GRADUATE SCHOOL OF OCEANOGRAPHY
UNIVERSITY OF RHODE ISLAND
NARRAGANSETT, RHODE ISLAND

Mooring Motion Correction of SYNOP Central Array Current Meter Data

Meghan Cronin,
Karen Tracey
and D. Randolph Watts

1992

GSO Technical Report No. 92-4

Abstract

From May of 1988 to August 1990, as part of the SYNOP field program, twelve tall moorings measured the Gulf Stream's temperature and velocity fields at nominal depths of 400 m, 700 m, 1000 m, and 3500 m. Although stiff, high-performance moorings were used to maintain the top current meters at approximately 400 m below the surface (~ 4000 m above the sea floor), the jet's drag caused the moorings to make vertical excursions.

Therefore, the current meter data were corrected to constant horizons using a modified version of Hogg's (1991) mooring motion correction scheme. An important extension of Hogg's (1991) method is the inclusion of a weighted interpolation of the measured temperatures. This modification assures that as the current meter measurements approach the respective nominal depths, the corrected temperature and velocity outputs smoothly approach the measurements; i.e. the compensated u, v, T records are truer to the measured records.

This report documents the mooring motion correction of the SYNOP Central Array temperature and velocity data.

This research was sponsored by the National Science Foundation contract #OCE97-17144 and the Office of Naval Research contract #s NOO014-90J-1568 and NOO014-90J-1548

Contents

List of Tables

List of Figures

1 Introduction

In the region between Cape Hatteras and the Grand Banks, the Gulf Stream is a strong and coherent jet with instantaneous speeds of up to 2 m s^{-1} near the surface and up to 0.2 m s^{-1} near the bottom. As the Gulf Stream flows in deep water near 68°W, the jet experiences large amplitude meanders, often forming and interacting with rings. SYNOP (**SY**Noptic **O**cean **P**rediction) is a multi-investigator research project, involving modelers, theoreticians, and observationalists, whose goals are to understand and model the dynamics governing the Gulf Stream meandering.

The SYNOP field program consisted of three arrays: an Inlet Array near Cape Hatteras, a Central Array near 68°W, and an Eastern Array just west of the Grand Banks near 55°W. The focus of this report is the Central Array, consisting of twelve tall, high-performance moorings which measured the Gulf Stream's temperature and velocity fields at nominal depths of 400 m, 700 m, 1000 m and 3500 m. In addition, Acoustic Doppler Current Profilers (ADCPs) were placed atop three of the moorings (I2, H3, and H4). Inverted echo sounders (IESs) with pressure sensors were placed near the base of each current meter mooring. The IES, ADCP and current meter sites in the Central Array are shown in Figure 1. Although most moorings had two deployment periods between May 1988 and August 1990, four of the tall moorings were in place for the full two-year period. During the second year, an additional thirteenth mooring, M13, was deployed. The current meter measurements are documented in Shay *et al.*, 1993.

Although fairing on the mooring wire and extra flotation were used to keep the moorings taut and maintain the top current meters at depths of approximately 400 m below the surface ($\sim$ 4000 m above the sea floor), the jet's drag caused the upper 1000 m of the moorings to make vertical excursions. Therefore, the current meter data was corrected to constant horizons using Hogg's (1991) mooring motion correction scheme. This report documents the mooring motion correction of the SYNOP Central Array's temperature and velocity data.

In the next section, Hogg's (1991) method will be briefly reviewed. In our application of Hogg's method, we made a slight modification to the temperature correction. This modification and the specific steps involved in correcting the SYNOP Central Array data set are discussed in Section 3. Extensive tests of the corrections are discussed in Section 4. Sections 5 and 6 show how we estimated the errors in the corrected temperature and velocity fields. Section 7

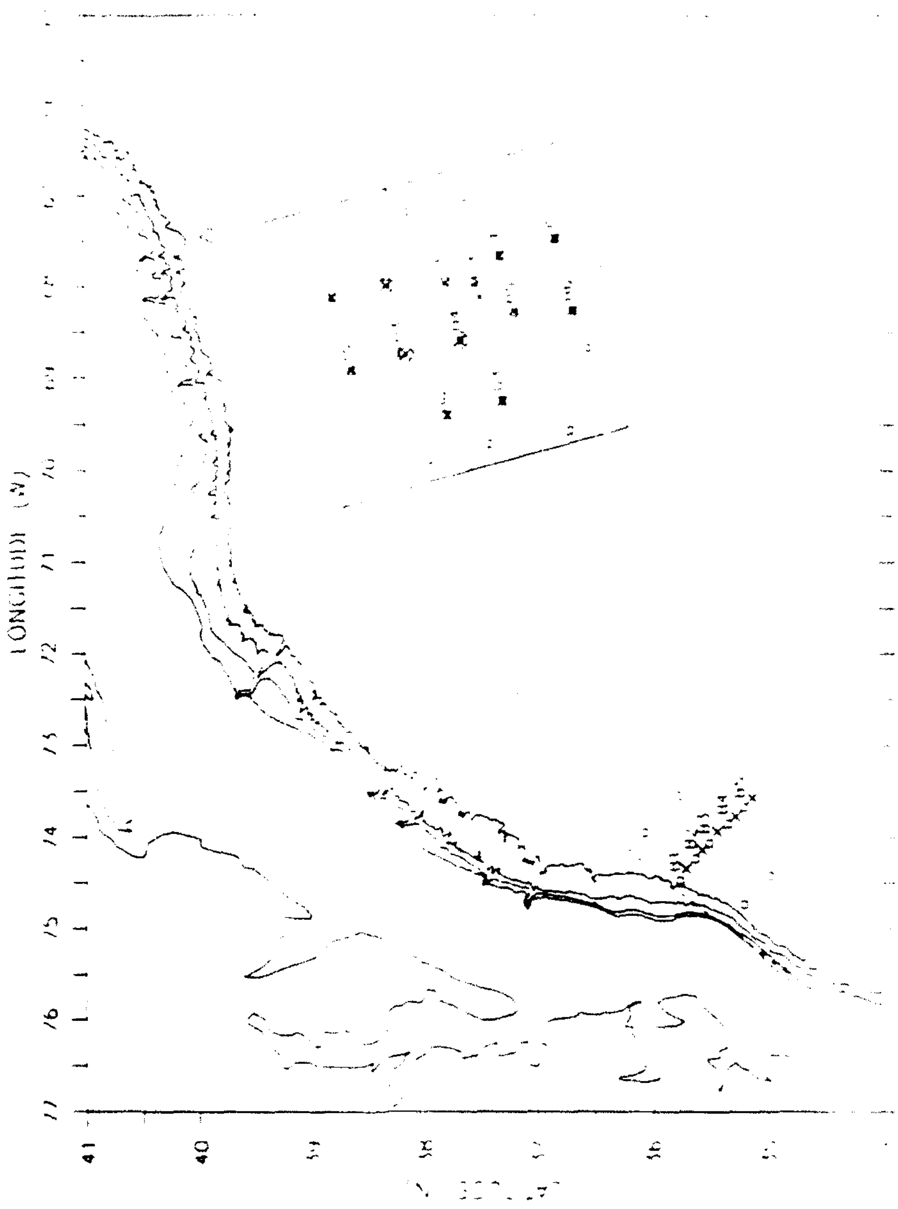


Figure 1: The SYNOP Central Array, centered near 38N/68W, is composed of twenty four IESs (boxes) and thirteen tall current meter moorings (x's). The IES at the base of each tall current meter moorings has a pressure sensor. Sites i2, h3, and h4 also have ADCPs (circles) atop the tall moorings.

discusses some useful byproducts of the mooring motion scheme, including the pseudo-IES and computation of the Brunt-Vaisala frequency.

2 Hogg (1991) Mooring Motion Correction Scheme

Hogg's (1991) mooring motion correction scheme assumes that all isotherms are parallel in a Gulf Stream cross-section. This is equivalent to assuming that the vertical profile of temperature has a 'canonical shape' at all times and locations: the profile is only shifted up and down as the Gulf Stream shifts back and forth across the mooring. The functional form used to describe the canonical temperature profile is a Nth order polynomial of the form:

$$T(x,p,t) = F(p_{ref}(x,t) - p) \tag{1}$$

$$F(p_{ref} - p) = 12°C + \sum_{n=1}^{N} c_n \, (p_{ref} - p)^{N+1-n} \tag{2}$$

The coefficients of the polynomial, c_n, are determined by performing a least-squares regression on the observed (T,p) data.

Once the coefficients have been determined, the canonical profile is shifted to fit the (T,p) measurements on a given mooring, yielding a time series of p_{ref} for that site. If the mooring consists of more than one current meter, the (T,p) pairs are regressed on the canonical profile to determine the optimal p_{ref}. Subsequently, the corrected temperatures at the desired pressure levels are obtained simply by

$$T_{cor}(p_{nom}) = F(p_{ref}(x,t) - p_{nom}). \tag{3}$$

To correct the current meter velocity measurements for mooring motion, the velocity is interpolated using temperature. The first step is to use the rotation matrix, $\mathbf{R}$, to rotate the velocity components from east-north coordinates to stream-coordinates. After correcting for mooring motion, the velocities are rotated back.

$$[v_s\hat{s},\ v_n\hat{n}]' = \mathbf{R}\left[u\hat{i},\ v\hat{j}\right]' \tag{4}$$

$$\left[u\hat{i}, v\hat{j}\right]' = \mathbf{R}^T\left(\mathbf{R}\left[u\hat{i},\ v\hat{j}\right]'\right) \tag{5}$$

$$= \mathbf{R}^T\,[v_s\hat{s},\ v_n\hat{n}]' \tag{6}$$

where

$$\mathbf{R} = \begin{bmatrix} cos\theta & sin\theta \\ -sin\theta & cos\theta \end{bmatrix} \tag{7}$$

and $\theta = atan((v_u - v_l)/(u_u - u_l))$. Note that $\mathbf{R}\,\mathbf{R}^T$ equals the identity matrix.

Assuming thermal wind and a well defined T-S relationship, the vertical change in the velocity can be related to the cross-stream temperature gradient. In stream coordinates, where the downstream (or shear) component of velocity is $v_s\,\hat{s}$, we obtain

$$\frac{\partial v_s}{\partial p} = +\frac{g\alpha}{f}\frac{\partial T}{\partial n} \tag{8}$$

$$= -\frac{g\alpha}{f}\frac{\partial p_{ref}}{\partial n}\frac{\partial F}{\partial p} \tag{9}$$

where α is the effective thermal coefficient of expansion, f is the Coriolis parameter and g is gravity. By integrating with respect to pressure, it can be shown that the change in velocity is proportional to the change in temperature:

$$\int_{p_u}^{p_l} \frac{\partial v_s}{\partial p}\partial p = \int_{p_u}^{p_l} -\frac{g\alpha}{f}\frac{\partial p_{ref}}{\partial n}\frac{\partial F}{\partial p}\partial p \tag{10}$$

$$v_s(p_l) - v_s(p_u) = -\frac{g\alpha}{f}\frac{\partial p_{ref}}{\partial n}[T(p_l) - T(p_u)]. \tag{11}$$

where the subscripts u and l refer to upper and lower depths. Thus,

$$\frac{v_s(p_{nom}) - v_s(p_l)}{T(p_{nom}) - T(p_l)} = \frac{v_s(p_l) - v_s(p_u)}{T(p_l) - T(p_u)} \tag{12}$$

Or.

$$v_s(p_{nom}) = \frac{v_s(p_l) - v_s(p_u)}{T(p_l) - T(p_u)}[T(p_{nom}) - T(p_l)] + v_s(p_l) \tag{13}$$

$$= m\,[T(p_{nom}) - T(p_l)] + v_s(p_l) \tag{14}$$

The cross-stream component of the velocity v_n must then be added to the corrected shear component $v_s(p_{nom})$ to obtain the corrected velocity vector, $\mathbf{U_{cor}}$:

$$\mathbf{U_{cor}} = v_s(p_{nom})\,\hat{s} + v_n\,\hat{n} \tag{15}$$

If the vertical shear is purely due to thermal wind, then m in Equation 4 is constant throughout the water column (for a given time) and the choice of levels u and l is arbitrary. However, it is advantageous to use the two current meters that are nearest in temperature to the corrected temperature: In the event that the measured temperature equals the corrected temperature, the corrected velocity will equal the measured velocity.

It should be noted however, that in Hogg's correction scheme this same principle does not apply to the temperature correction. In the event that the current meter is at the nominal pressure, the corrected temperature, obtained from the canonical profile, is not necessarily the measured temperature. As described in the next section, in our application of Hogg's correction scheme to the SYNOP Central Array data, we modified the temperature correction procedure to require that $T_{cor}(p_{nom}) = T_p$ when $p = p_{nom}$.

3 Application to the SYNOP data

3.1 The SYNOP Central Array measurements

Typically, the vertical excursions of the SYNOP current meter moorings were on the order of 50 meters. Occasionally though, the excursions were larger. For example, one large excursion taken by mooring H6 exceeded 550 m.

The moorings were designed to have their upper 1000 meters remain essentially vertical at all times. Additionally, fairing was installed on the wire between the three top current meters to improve the performance. Table 1 summarizes the conditions of the fairing upon recovery. As noted in that table, fish nets were tangled on some moorings. However, the nets were always near the bottom current meter and therefore did not significantly affect the mooring motion.

For most moorings, the typical pressure differences between the level 1 and level 2 current meters were 3060 kPa (303 m[1]). Between level 1 and level 3 the typical delta pressures were 2.02×3060 kPa (612 m). However, due to differences in flow conditions (e.g. strong, moderate, or weak currents) and differences in the buoyancy and drag (fairing) of each mooring, the actual delta pressures vary from mooring to mooring. Table 2 lists the differences between the measured pressures by the current meters at levels 1 and 2 (and between levels 1 and 3 where available). First order statistics on the vertical excursion of each mooring are also given in Table 2.

Each of the three moorings prepared by the University of Miami (H3, H4, and I2) had an ADCP, with pressure and temperature sensors, mounted 12 m above the top current meter (Figure 2). The ADCPs measured the velocities throughout the upper 400 m of the water column. To reduce noise, the velocities are averaged within 9 m bins. As shown in Figure 2,

[1]Note that 1 m = 1.01 db = 10.1 kPa.

Table 1. Comments on Condition of Moorings from Recovery Logbooks

The upper three current meters on each mooring were separated by 300 m pieces of wire. These two sections are designated as L1 and L2. Fairing, in 1-5 ft lengths, was installed on both the L1 and L2 wire lengths.

YEAR 1	
G2	Two 2-3 ft pieces of fairing were stuck on L1.
G3	OK
H3	Eight 5 ft pieces of fairing jammed on L1.
H4	Three or four 1-5 ft pieces of fairing jammed on L1.
H5	Some pieces of fairing were jammed on both L1 and L2.
I2	Two 5 ft pieces of fairing jammed on L1.
I3	OK
I4	OK
YEAR 2	
G2	One piece fairing not spinning freely. One piece of fairing is jammed.
G3	Damaged fairing: about 10 pieces were broken, cut, or jammed.
H2	Two pieces of fairing on L1 and two pieces on L2 were cocked.
H3	The pieces of fairing on L1 was jammed together, but were spinning freely. 8 glass balls (above bottom VACM) were tangled in 1m by 6m fishnet. 1 glass ball imploded at bottom.
H4	2 m of jammed fairing on L1. Snagged net at the connection between the two 500m sections above bottom VACM (2000m below the level 3 current meter).
H5	One piece of jammed fairing.
H6	6 m of fish net were tangled somewhere between 200-700 m above bottom VACM. 1 glass ball imploded (3rd from bottom).
I1	Three pieces of fairing were jammed and cocked on L1. About 20% of fairing was cocked, but this probably happened on recovery.
I2	10 m of fairing jammed together, but these were spinning freely.
I3	OK
I4	OK
I5	OK
M13	Two pieces of jammed fairing. Two of 16 glass balls imploded near bottom.

Table 2. Current Meter Pressure Statistics

First order statistics of the level 1 current meter pressures in the SYNOP Central Array are listed. Also tabulated are the means and the standard deviations of the pressure differences between the level 1 and levels 2 and 3 sensors. The pressure sensors, at nominal depths of 400 m, 700 m, and 1000 m, are respectively denoted as P1meas, P2meas, and P3meas. Pressures are expressed in units of 1000 kPa (or 100 db). The symbol "NA" indicates no data.

YEAR 1

	P2meas - P1meas		P3meas - P1meas		P1meas			
Site	Mean	Std	Mean	Std	Mean	Min	Max	Std
G2	3.043	0.039	NA	NA	3.572	3.200	4.663	0.371
G3	NA	NA	NA	NA	4.052	3.470	5.116	0.448
H3	3.131	0.008	6.401	0.014	3.259	3.226	3.695	0.068
H4	3.044	0.035	6.221	0.085	3.989	3.483	5.464	0.363
H5	NA	NA	NA	NA	3.846	3.336	6.414	0.609
I2	3.207	0.014	NA	NA	3.389	3.363	3.654	0.042
I3	NA	NA	NA	NA	3.755	3.176	5.956	0.481
I4	3.085	0.028	NA	NA	3.743	3.188	5.409	0.507

YEAR 2

	P2meas - P1meas		P3meas - P1meas		P1meas			
Site	Mean	Std	Mean	Std	Mean	Min	Max	Std
G2	NA	NA	NA	NA	3.845	3.396	5.318	0.408
G3	NA	NA	NA	NA	3.957	3.547	6.023	0.460
H2	3.077	0.003	NA	NA	3.720	3.664	4.602	0.112
H3	3.122	0.050	NA	NA	3.360	3.127	4.656	0.291
H4	3.009	0.018	NA	NA	3.890	3.547	5.669	0.351
H6	NA	NA	NA	NA	4.470	3.665	9.676	1.246
I1	3.093	0.004	NA	NA	3.775	3.723	4.584	0.107
I2	NA	NA	NA	NA	NA	NA	NA	NA
I3	3.046	0.038	6.250	0.079	3.599	3.136	5.804	0.567
I4	3.025	0.120	6.257	0.082	3.737	3.171	6.847	0.668
I5	3.101	0.012	NA	NA	4.301	3.627	8.373	0.936
M13	3.081	0.049	NA	NA	3.654	3.157	6.167	0.496

NOTE: For site I4 during Year 2, the standard deviation between levels 1 and 2 is greater than that between levels 1 and 3 due to a drift in the level 2 sensor. However this drift is of no consequence because the observed level 2 pressures are not used in the mooring motion correction.

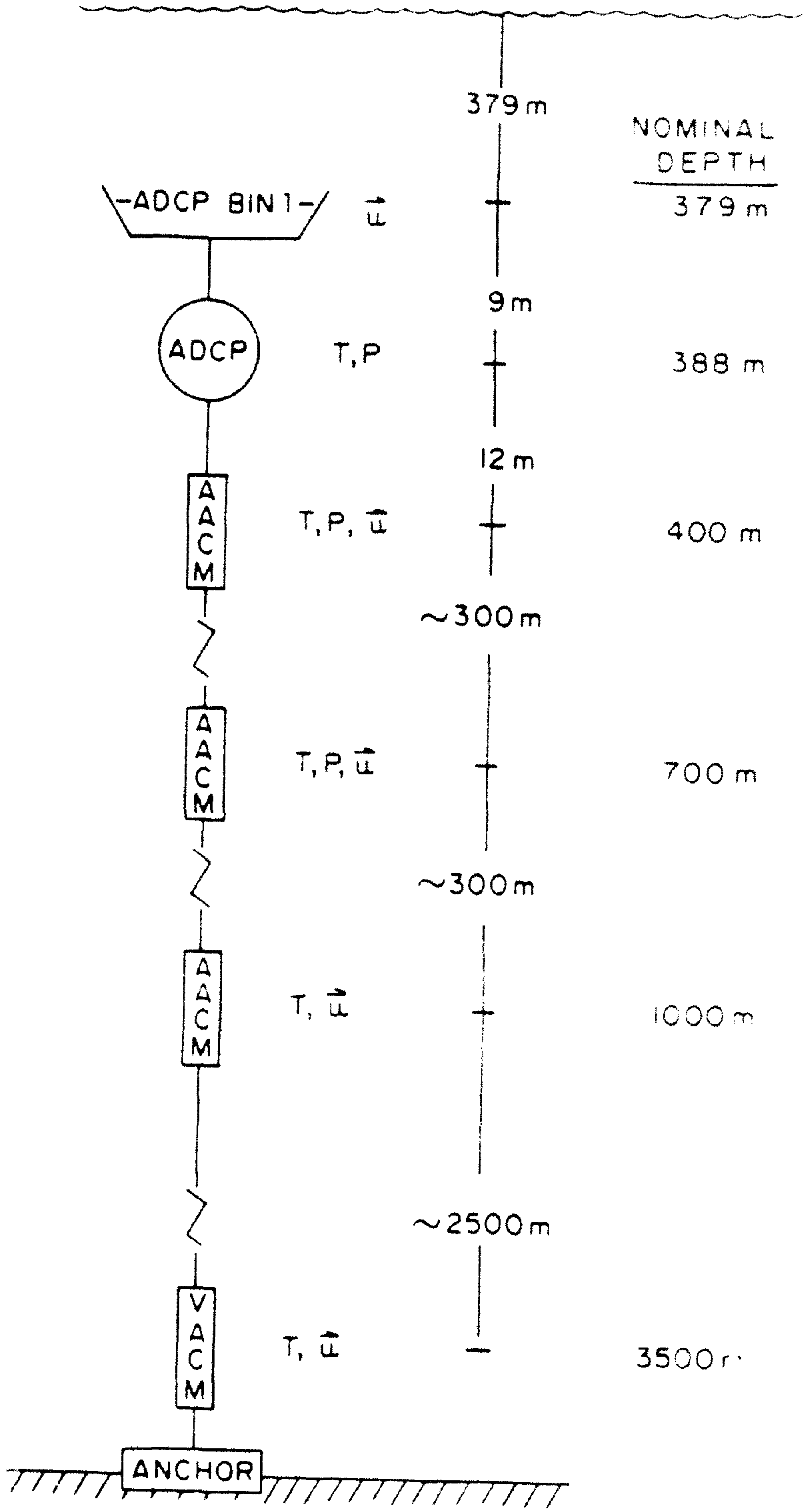


Figure 2: Schematic diagram of a tall current meter mooring.

the Bin 1 velocities are located 9 m above the ADCP itself and 21 db above the top current meter. Thus the ADCPs provided redundancy in the top level measurements.

Appendix A compares the temperatures measured by the ADCPs and the upper level current meters at four sites. For two of the sites, H3_YR2 and I2_YR2, there is good consistency between the measurements. This was not the case for two other sites, H3_YR1 and H4_YR1. However, the differences are not surprising because the accuracy of the ADCP temperature measurements is not as good as that of the current meters. Thus, the ADCP temperatures were not used in the mooring motion correction procedures except where the temperature sensors on the level 1 current meters failed.

On the other hand, the ADCP pressure measurements could be validated by acoustic tracking (B. Johns, pers. comm.) and were deemed to be more trustworthy than the current meter pressures. Thus the ADCP pressures were used whenever possible. First order statistics of the ADCP pressures are reported in Table 3 together with the mean pressure differences between the ADCP and the upper two current meters. Taking into account the wire lengths and mooring design, it was shown that there was a 6 db discrepancy between pressures measured by the ADCP and those of the top level current meters on all three Miami moorings. Comparisons with the acoustic tracking depths revealed that the current meter pressures were too large (most likely caused by Miami pressure calibration errors). Therefore, the ADCP pressures were used for the mooring motion correction with data gaps filled by the current meter pressures after subtracting the 6 db bias.

In order to correct the velocities and temperatures for mooring motion, the pressure of each current meter at the time the measurements were made must be known. However not all of the level 2 and level 3 current meters had pressure sensors, and furthermore, some of the measurements were questionable. For example, several of the level 2 current meters on the Miami moorings exhibited pressure biases similar to those found with the level 1 instruments. Based on the mooring design, the differences in pressure between the top three current meters were expected to be nearly constant despite the mooring motion. Thus, the top current meter pressures were used together with constant offsets to simulate the daily pressures at the level 2 and 3 instruments as

$$p2(t) = p1(t) + \mathrm{delp12} \tag{16}$$

$$p3(t) = p1(t) + \mathrm{delp13} \tag{17}$$

Table 3. ADCP Pressure Statistics

First order statistics of the ADCP pressures in the SYNOP Central Array are listed. Also tabulated are the means and the standard deviations of the pressure differences between the ADCP and current meters at levels 1, 2, and 3. The ADCP pressures, designated as *Pbin1*, correspond to the depth of the Bin 1 velocities (9 m above the ADCP or 21 m above the level 1 current meter). The current meter sensors, at nominal depths of 400 m, 700 m and 1000 m, are respectively denoted as P1meas, P2meas, and P3meas. Pressures are expressed in units of 1000 kPa (or 100 db). The symbol "NA" indicates no data.

YEAR 1										
	P1meas - Pbin1		P2meas - Pbin1		P3meas - Pbin1		Pbin1			
Site	Mean	Std	Mean	Std	Mean	Std	Mean	Min	Max	Std
H3	0.258	0.007	3.384	0.008	6.658	0.016	2.994	2.979	3.081	0.019
H4	0.280	0.014	3.309	0.037	6.442	0.089	3.831	3.239	4.448	0.336

YEAR 2										
	P1meas - Pbin1		P2meas - Pbin1		P3meas - Pbin1		Pbin1			
Site	Mean	Std	Mean	Std	Mean	Std	Mean	Min	Max	Std
H3	0.272	0.025	3.394	0.060	NA	NA	3.088	2.862	4.431	0.296
I2	NA	NA	3.216	0.035	NA	NA	3.251	3.034	5.065	0.314

The offsets delp12 and delp13 were determined for each mooring based on both the mooring design (wire lengths and stretching) and the observations.

Comparisons of the simulated and observed pressures were made by looking at the mean and extreme differences between the records. The results are summarized in Table 4. In general, the differences are under 10 db (0.10 kPa) as anticipated by the mooring design. The large mean differences on Miami moorings H3 and I2 are assumed to be related to calibration errors of the current meters since $p2$ and $p3$ are simulated from the acoustically-verified ADCP pressures. Table 4 also indicates long term drifts in the observed pressures. Several instruments had drift rates of about 4 db per year. While these drifts are too high to use the observed pressures for dynamical analyses, they are small enough that that do not significantly affect the mooring motion correction.

In the above equations, $p1$ is defined as the pressure at the upper most temperature measurement ($T1$). For the most part, $T1$ and $p1$ refer to the measurements made by the level 1 current meter (Table 5). However this is not true for M13 and the three Miami moorings. For site M13, the level 1 pressure sensor didn't function properly during a 50 d period. So instead, we used the level 2 current meter pressure record and chose the appropriate values for delp12 and delp13 (listed in Table 5) to determine the level 1 and level 3 pressures. For Miami moorings H3_YR1, H3_YR2, and H4_YR1, ADCP pressures were used instead of the current meter pressures. However, the level 1 current meter temperatures were still used as $T1$ for those moorings. Consequently, the ADCP pressures (Pbin1) needed to be adjusted by 21 m from the depth of the Bin 1 velocities to the depth of the level 1 current meter (Figure 2). Thus $p1 = P\text{bin}1 + 21$ db for those moorings. For Miami moorings H3_YR2 and I2_YR2, the ADCP temperatures were used as $T1$; thus the Pbin1 pressures were offset by 9 m depth to be the depth of the ADCP (Figure 2). Since the ADCPs failed on moorings I2_YR1 and H4_YR2, the current meter pressures were used for the mooring motion correction. However, as noted above, these needed to have a 6 db bias removed. Table 5 summarizes how $p1$ and $T1$ were determined for each mooring. The offset constants, delp12 and delp13, are also listed in Table 5.

The mooring motion scheme also requires that the upper most velocity ($U1$) and its pressure (PU) be specified. As indicated in Table 5, the level 1 current meter velocities were used in all but two cases. Thus, typically $PU = p1$. However when the ADCP velocities were used, $PU = P\text{bin}1 = p1 - 9$ db.

Table 4. Comparison of Simulated and Measured Pressures at Levels 2 and 3

P2 and P3 are simulated pressures from P1 using Equations 16 and 17. P2meas and P3meas are the observed pressures on the moorings. "Good agreement" indicates that the offsets and peak differences between the simulated and measured pressures fall within the ranges anticipated by the mooring design. Pressure units are kPa. A record length of 1400 pts corresponds to a period of about one year.

Mooring	Observed Drifts	P2 - P2meas Offset	P2 - P2meas Extremes	P3 - P3meas Offset	P3 - P3meas Extremes	Comments
G2_YR1	None	−0.025	0.15	NA	NA	Good agreement
H2_YR2	None	0.005	0.04	NA	NA	Good agreement
H3_YR1	P1: −0.03 over 1400 pts. P3meas: −0.03 over 300 pts.	−0.13	0.10	−0.23	0.09	P1 (ADCP) verified acoustically; offsets are due to current meter biases.
H3_YR2	P2meas: −0.12 over 1000 pts	−0.20	0.3	NA	NA	P1 (ADCP) verified acoustically; offsets are due to current meter biases.
H4_YR2	None	−0.04	0.10	NA	NA	Good agreement
I1_YR2	P1: −0.022 over 3200 pts. P2meas: −0.013 over 3200 pts.	None	0.04	NA	NA	Good agreement
I2_YR1	P1: 0.01 over 1500pts. P2meas: 0.07 over 700pts.	0.22	0.06	NA	NA	P1 (ADCP) verified acoustically; offsets are due to current meter biases.
I2_YR2	None	−0.07	0.11	NA	NA	P1 (ADCP) verified acoustically; offsets are due to current meter biases.
I3_YR2	None	−0.015	0.12	−0.15	0.2	Good agreement
I4_YR1	None	None	0.06	NA	NA	Good agreement
I4_YR2	P1meas: 0.05 over 1400 pts. P2meas: −0.5 over 600 pts	0.1	0.65	−0.10	0.22	P2meas has large drift
I5_YR2	P1: −0.04 over 3000 pts	−0.01	0.05	NA	NA	Good agreement
M13_YR2	P2meas: −0.06 over 1400pts	None	0.5	NA	NA	Level 1 pressure missed several mooring excursions. Use P2meas to simulate pressures of levels 1 and 3.

Table 5. Data Sources of the Top Level Temperatures, Pressures, and Velocities used in the Mooring Motion Correction

Pressures are expressed in units of decibars. The constant offsets, delp12 and delp13, were used to simulate P2 and P3 respectively from P1 according to Equations 16 and 17. The university technical group that prepared each mooring is indicated. See the text and Appendix E for further explanations for each site.

T1 = Measured temperature at top level
U1 = Measured velocity at top level
P1 = Measured pressure at top temperature
PU = Pressure at top velocity
CM1 = Top level is level 1 current meter
CM2 = Top level is level 2 current meter
ADCP = Top level temperature is ADCP (12 m above CM1)
Bin1 = Top level is Bin 1 (9 m above ADCP: 21 m above CM1)

YEAR 1

Mooring	Group	T1	U1	P1	delp12	delp13	PU-P1
G2	URI	CM1	CM1	PCM1	304	2.03*304	0
G3	URI	CM1	CM1	PCM1	306	2.03*306	0
H3	MIAMI	CM1	CM1	Pbin1+21	306	2.04*306	0
				PCM1−6			
H4	MIAMI	CM1	CM1	Pbin1+21+10	306	2.03*306	0
H5	URI	CM1	CM1	PCM1	306	2.03*306	0
I2	MIAMI	CM1	CM1	PCM1−6	305	2.04*305	0
I3	URI	CM1	CM1	PCM1	306	2.03*306	0
I4	URI	CM1	CM1	PCM1	308	2.03*308	0

YEAR 2

Mooring	Group	T1	U1	P1	delp12	delp13	PU-P1
G2	URI	CM1	CM1	PCM1	306	2.03*306	0
G3	URI	CM1	CM1	PCM1	306	2.03*306	0
H2	WHOI	CM1	CM1	PCM1	308	2.04*308	0
H3	MIAMI	CM1	CM1	Pbin1+21	306	2.03*306	0
		ADCP	Pbin1	Pbin1+9			−9
H4	MIAMI	CM1	CM1	PCM1−6	304	2.03*304	0
H6	WHOI	CM1	CM1	PCM1	309	2.02*309	0
I1	WHOI	CM1	CM1	PCM1	309	2.04*309	0
I2	MIAMI	ADCP	Pbin1	Pbin1+9	306+12	(2.04*306)+12	−9
I3	URI	CM1	CM1	PCM1	305	2.02*305	0
I4	URI	CM1	CM1	PCM1	306	2.04*306	0
I5	WHOI	CM1	CM1	PCM1	310	2.02*310	0
M13	URI	CM2	CM2	PCM2	−308	1.03*308	0

3.2 STEP 1: Determine the canonical profile

Hogg's mooring motion correction method relies upon the assumption that the isotherms are parallel and therefore a canonical temperature profile exists. Although this assumption is generally valid below 16°C, it is not necessarily true for warmer waters, especially near 18°C. Because the 16°C isotherm is typically found at depths above the uppermost current meter (400 m) across most of the Gulf Stream, this assumption is nearly valid for all SYNOP current meter moorings. Despite this, we found that it was best to apply different profiles to the northern and southern moorings. To create these profiles, the moorings were separated into two groups. For each region, the data were then strung together to create a single long $p1$ time series and corresponding time series of temperature for all three levels. Additionally, long time series of $p2$ and $p3$ were determined according to Equations 16–17 using representative values for delp12 and delp13. Separate profiles for the northern and southern regions were subsequently determined by least squares regression. (The MATLAB codes are given in Appendix B.)

Temperature data from the first year of moorings H3 and I3, and the second year of moorings H2, I1, I2, and H3 were used to determine the northern profile. The criteria used to select those sites were as follows: (1) The mooring must have at least two working current meters (Figure 3 shows the data recovery for each current meter). (2) The separation between the level 1 and level 2 current meters on the mooring must be 3070 kPa ± 30 kPa, and it must be 6250 kPa ± 50 kPa between the level 1 and level 3 instruments. (3) For most of the time, the mooring should be located north of the north wall but not in the recirculation region.[2] (4) Changes in the canonical profile caused by the inclusion of the data from that site improves the motion corrected data, as indicated by tests such as those described below in Section 4. The northern profile, shown in Figure 4, is a 7th order polynomial whose coefficients are listed in Table 6. The data on moorings H2, H3, I1, I2 for both years, and the first year of data on moorings G2 and I3, and the second year of data on mooring H4 were corrected using the northern profile.

Similarly, the temperature data from year 1 of moorings H3, I4, and I5, and data from year 2 of moorings H3, I4, M13, and I5 were used to determine the southern profile. Again, the criteria for choosing these sites were: (1) The moorings must have at least two working current meters. (2) The separation between levels 1 and 2 must be 3075 kPa ± 50 kPa, and

[2] During the first year, the H2 and I1 moorings were in the recirculation region and their data has been exluded from the determination of the northern profile.

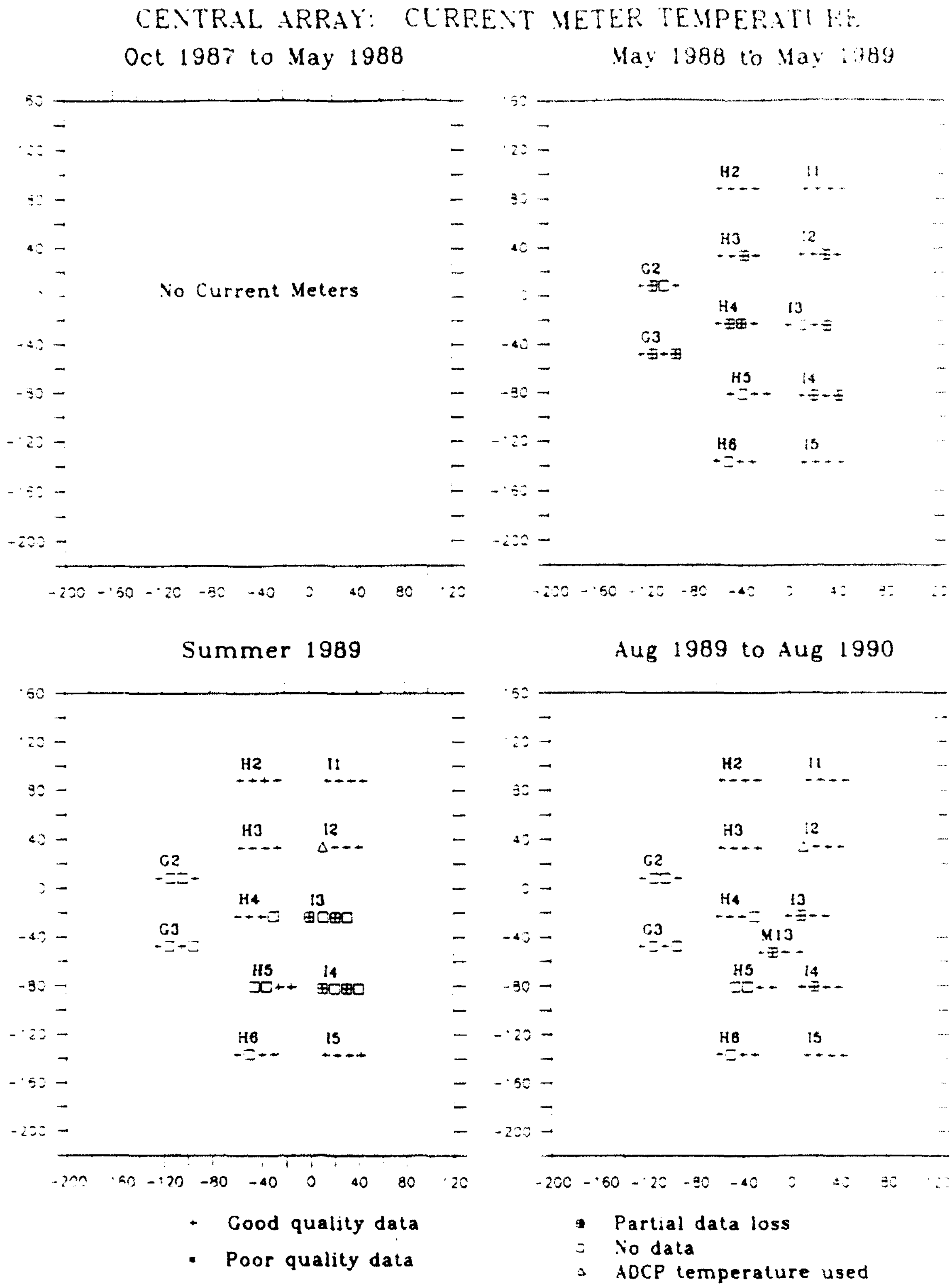
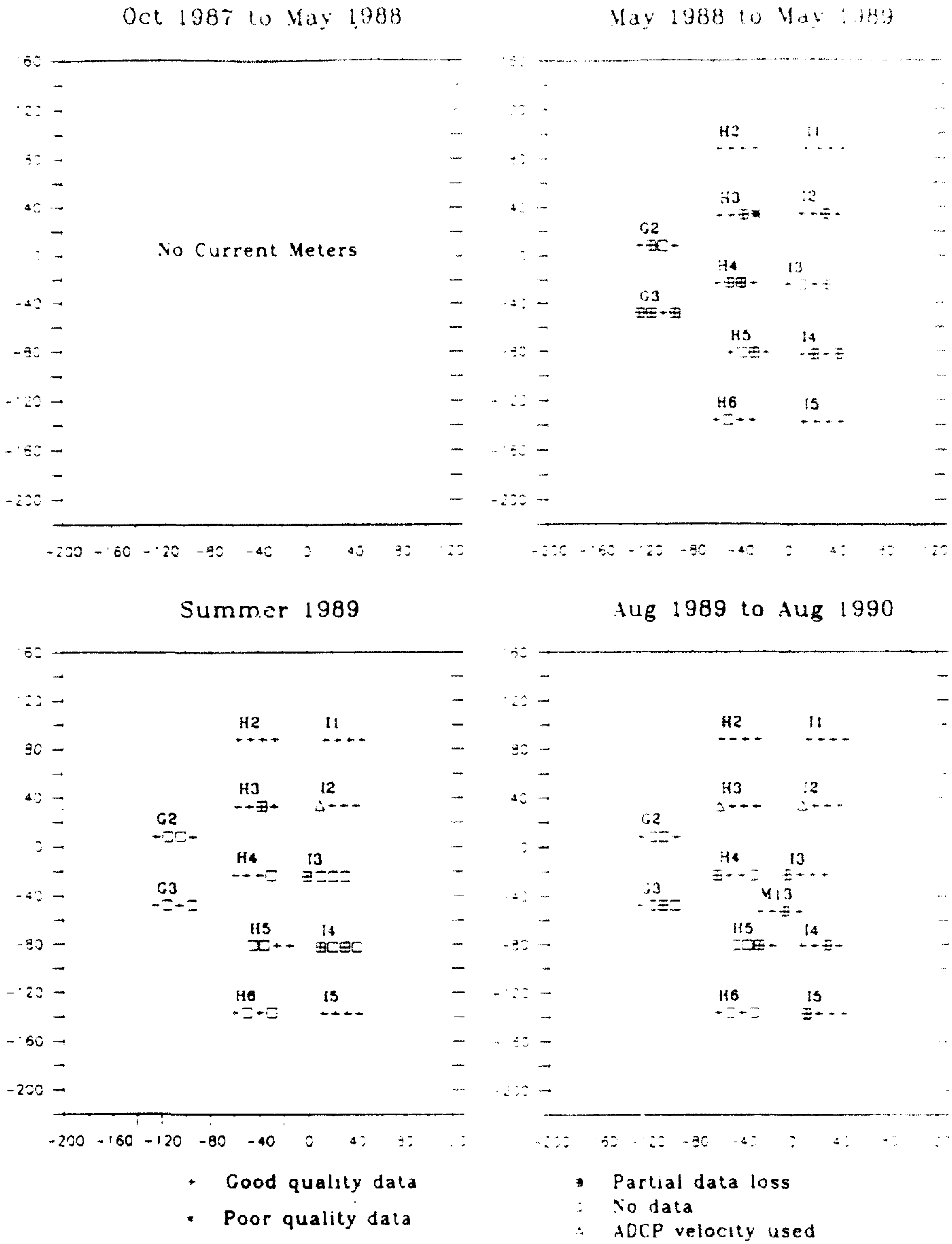


Figure 3: Current Meter temperature and velocity data returns during the two-year deployment period from May 1988 to August 1990. The four symbols indicate the data recovery for the current meters at levels 1–4 on each mooring, where the left symbol corresponds to the level 1 instrument. The axes labels are in kilometers from the origin at 38°N, 68°W, with the x-axis rotated to be oriented along 075°T.

CENTRAL ARRAY: CURRENT METER VELOCITY



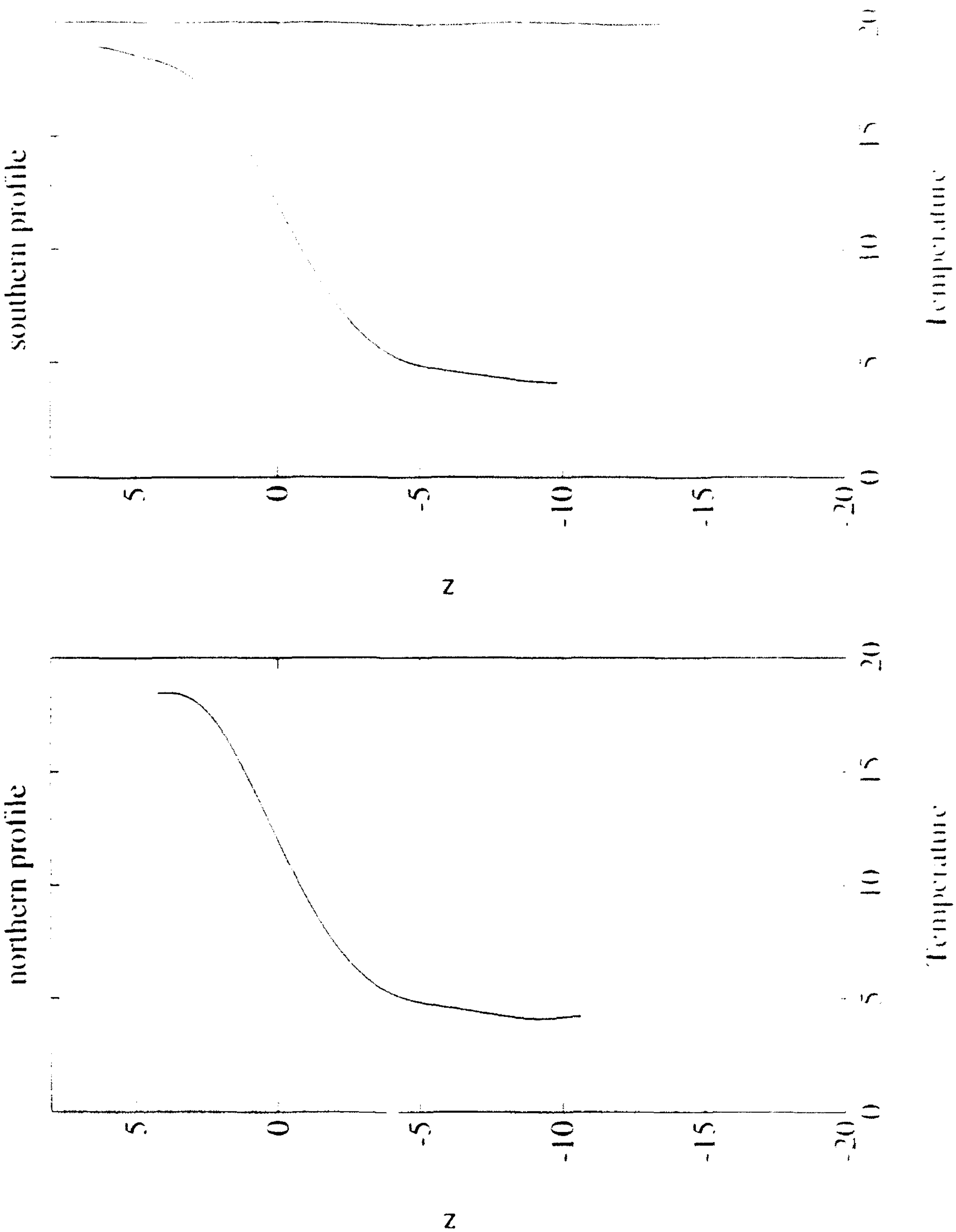


Figure 4: Northern and Mid-stream/Southern canonical temperature profiles, $T = F(z)$, where $z = p_{ref} - p$.

Table 6. Coefficients for the Northern and Southern Canonical Profiles.

The southern profile is a 9th order polynomial in z where $z = p_{ref} - p$ (in Pa). The northern profile is a 7th order polynomial. The coefficients were found by least squares regression. The zeroth order coefficient was set to be 12. Thus when $p = p_{ref}$, $T = 12°C$.

n	Southern Profile Coefficients	Northern Profile Coefficients
9	−2.2649178e−08	0.0000000e+00
8	−9.3254665e−07	0.0000000e+00
7	−8.4516302e−06	6.9965043e−06
6	7.9955372e−05	1.8593337e−04
5	1.2677613e−03	1.1301407e−03
4	−1.8511301e−03	−7.5093500e−03
3	−6.7733505e−02	−8.5146575e−02
2	−1.0571906e−03	6.1521248e−02
1	2.4092789e+00	2.6986492e+00
0	12	12

between levels 1 and 3 it must be 6240 ± 30 kPa. (3) The mooring must generally be positioned either mid-stream or on the southern side of the north wall. (4) The inclusion of the data from that site to define the profile enhances the overall motion correction as indicated by the tests described in Section 4. For example, although mooring H3 was corrected using the northern canonical profile, it was found that the inclusion of H3 in determining the southern canonical profile helped improve the correction of moorings I3 and M13. The southern profile, shown in Figure 4, was used to correct both years of data from moorings G3, H6, I4, I5, and the first year of data from H4 and H5, and the second year of data from moorings G2, M13, and I3. The coefficients for the southern profile, a 9th order polynomial, are listed in Table 6.

3.3 STEP 2: Correct the temperature data on a given mooring

3.3.1 STEP 2a: Determine reference pressure

The first step in correcting the temperature is to determine the reference pressure, p_{ref}. The reference pressure is defined as the pressure of the 12°C isotherm, as specified by the zeroth order coefficients of the northern and southern profiles (Table 6). We solve for p_{ref} by minimizing $\sum_{k=1}^{N}[T_k - F(p_{ref} - p_k)]$ for the temperature and pressure (T, p) measurements at the two or three (N) current meters on each mooring. The polynomial $T = F(p_{ref} - p)$ is specified to be either the northern or southern profile, depending on the criteria listed previously. The minimization is performed for each sample period, producing a time series of p_{ref} for each mooring. How well this minimization procedure works for each mooring can be ascertained by the plots of measured temperatures versus $p_{ref} - p$ shown in Appendix C.

There were a few moorings which required special treatment. The top current meter on mooring I2_YR2 failed; this is a critical instrument for determining p_{ref} because it was located in the high gradient portion of the canonical profile. Fortuitously, there was a working ADCP located 12 m above the current meter. Thus we were able to use the ADCP temperatures and pressures for the regression to determine p_{ref}. At two other sites, H4_YR1 and G2_YR2, the moorings had only one working temperature sensor, making it impossible to determine p_{ref} by regression. However we were able to obtain p_{ref} at those sites from a different data source. As part of the Central Array, IESs were located near the base of each mooring. The IESs measure the depth of the thermocline as indicated by the 12°C isotherm. Thus, the Z_{12} measured by the IESs is equivalent to the reference pressure (after taking into account the unit conversions from

depth to pressure. $p_{ref} = 1.01 * Z_{12}$) because by definition. $T = 12$ when $p = p_{ref}$. However, there was a 2-5 km distance separating the current moorings and the IESs. This separation was considered too far to use the IES Z_{12} measurements directly in the mooring motion correction. Instead, we interpolated objectively-analyzed maps of Z_{12} (Tracey and Watts, 1991) to obtain time series of p_{ref} right at the two mooring sites. At mooring H4_YR1, a second current meter worked for half of the deployment year. Thus we were able to obtain a partial record of p_{ref} by regressing the current meter data. A comparison of the regressed p_{ref} to that of the IES revealed a bias of 10 db between the records. Thus in order to make the records consistent, the 10 db offset was added to the Pbin1 pressures record of site H4_YR1 before correcting the temperatures.

Not only did we interpolate the IES maps to sites H4_YR1 and G2_YR2, we also interpolated them to obtain Z_{12} records at all the moorings. Plots of p_{ref}, determined from the current meters and scaled into depth in meters, are shown together with the Z_{12} records from the IESs in Appendix D for all the moorings. The agreement between the two types of records is quite good, with rms differences between the two records generally less than 25 m.

3.3.2 STEP 2b: Correct the temperature data

Once p_{ref} is obtained for a given mooring, the temperatures at the desired pressure levels can be determined by using Hogg's method (Equation 3) and specifying the appropriate canonical profile. However we modified Hogg's method slightly by using a weighted average of the temperature correction from the two nearest temperature-pressure pairs, (T_u, p_u) and (T_l, p_l). That is,

$$T_{cor} = w_u T_u + w_l T_l \tag{18}$$

$$\text{where,} \quad T_u = F(p_{ref} - p_{nom}) + [T(p_u) - F(p_{ref} - p_u)] \tag{19}$$

$$\text{and,} \quad T_l = F(p_{ref} - p_{nom}) + [T(p_l) - F(p_{ref} - p_l)] \tag{20}$$

$$\text{while,} \quad w_u = \frac{|p_l - p_{nom}|}{|p_{nom} - p_u| + |p_{nom} - p_l|} \tag{21}$$

$$\text{and,} \quad w_l = \frac{|p_{nom} - p_u|}{|p_{nom} - p_u| + |p_{nom} - p_l|} \tag{22}$$

The weights sum to 1 and are linearly proportional to the pressure differences of the measurements away from p_{nom}; the T_u and T_l differ from $T_{nom} = F(p_{ref} - p_{nom})$ by the measured

temperature differences at the respective levels. This modification forces the corrected temperature to smoothly approach and agree with the measured temperature when the current meter pressure approaches and equals the nominal pressure: i.e. when $p_l = p_{nom}$, $T_{cor} = T_l$.

On the moorings not used to determine the canonical profiles (Appendix E), it is possible for the current meters to be deeper or shallower than the canonical profile. i.e. either $p_{ref} - p_l$ or $p_{ref} - p_l$ lies beyond the range for which the polynomial $T = F(p_{ref} - p)$ has been defined. Under those conditions, the temperatures are not corrected using Equation 18, but instead are corrected using Equation 3, where $T_{cor} = F(p_{ref} - p_{nom})$. If $p_{ref} - p_{nom}$ is also beyond the range of the canonical profile, then the temperature cannot be corrected. The pressure ranges for the northern and southern profiles can be found in the first column of Table 8.

3.4 STEP 3: Correct the velocity data

To correct the velocity data of a given current meter, the data from two *nearest* current meters are used. These current meters are selected based on their *temperatures*: the ones closest in temperature to the corrected temperature are chosen. First, the velocities are decomposed into cross-stream and downstream components as in Equation 4. Next, the downstream component is linearly interpolated to the corrected temperature according to Equation 13. Subsequently, the cross-stream component is added back to the corrected downstream component to obtain the corrected velocity (Equations 6 and 15). Note that with our modification to Hogg's temperature correction, if one of the current meters is at the nominal pressure, the corrected temperature is the measured temperature and the corrected velocity is the measured velocity. If the current meter is not at the nominal pressure, the corrected velocity is a smoothly varying function between the two measurements.

If the temperature sensor on a given current meter did not work, the mooring motion temperature correction scheme was used to simulate temperature data at the current meter's *observed* pressure. Subsequently, the simulated temperature was used as either $T(p_l)$ or $T(p_u)$ in Equation 13 to correct the measured velocities. If there was only one velocity measurement among the upper three levels, the deep (3500 m level) velocity and temperature measurements were used as $(T(p_l), u(p_l), v(p_l))$ in Equation 13.

On moorings I2_YR2 and H3_YR2, no velocity data were obtained by the level 1 current meters. However, we were able to use the ADCP Bin 1 velocities to fill those gaps. To correct

the ADCP velocities to the nominal pressure level according to Equation 13, the ADCP temperatures first had to be corrected to the Bin 1 level, located 9 m above the ADCP instrument itself.

A summary, mooring-by-mooring, of all the mooring motion compensation procedures is given in Appendix E. The corrected temperature and velocity data are shown in Appendix F.

4 Tests of the Corrections

We now present the results of extensive testing that show that this correction scheme is very robust. In particular, two tests will be discussed. The first test examines the ability of the correction scheme to simulate data by interpolating between two current meters, and the second test evaluates extrapolation. Each test was applied to both a northern mooring (I1) and a southern mooring (I4_YR2); these moorings were selected because all three current meters worked properly.

For 'Test 1', level 1 and level 3 current meters are used to interpolate to level 2, which is approximately 300 m away from either input. The simulated level 2 current meter temperature and velocity data, as well as covariances and heat fluxes, are then compared to the directly measured level 2 data. The comparisons are shown in Figure 5 and the rms errors are listed in Table 7. The simulated and observed velocities exhibit rms differences of under 6 cm s^{-1}, which is quite small considering the large 600 m distance between the level 1 and level 3 current meters.

An even more rigorous test involves an extrapolation. For 'Test 2', level 2 and level 3 current meters are used to <u>extrapolate</u> up to level 1. Again, the simulated level 1 temperature and velocity data, as well as covariances and heat fluxes, are then compared to the measured level 1 current meter data. The simulated and the observed time series are shown in Figure 6 and the rms errors are listed in Table 7. The velocities for the northern mooring have rms differences of under 8 cm, while those for the southern mooring are twice as large. The larger errors for the southern mooring can be attributed to the deeper and more frequent vertical excursions taken by the mooring because it was located in a higher-velocity region of the current. Considering the large extrapolation distances (the level 1 current meter is 300 m and 600 m away from the level 2 and level 3 instruments, respectively) the observed errors are small and indicate that the correction scheme is robust.

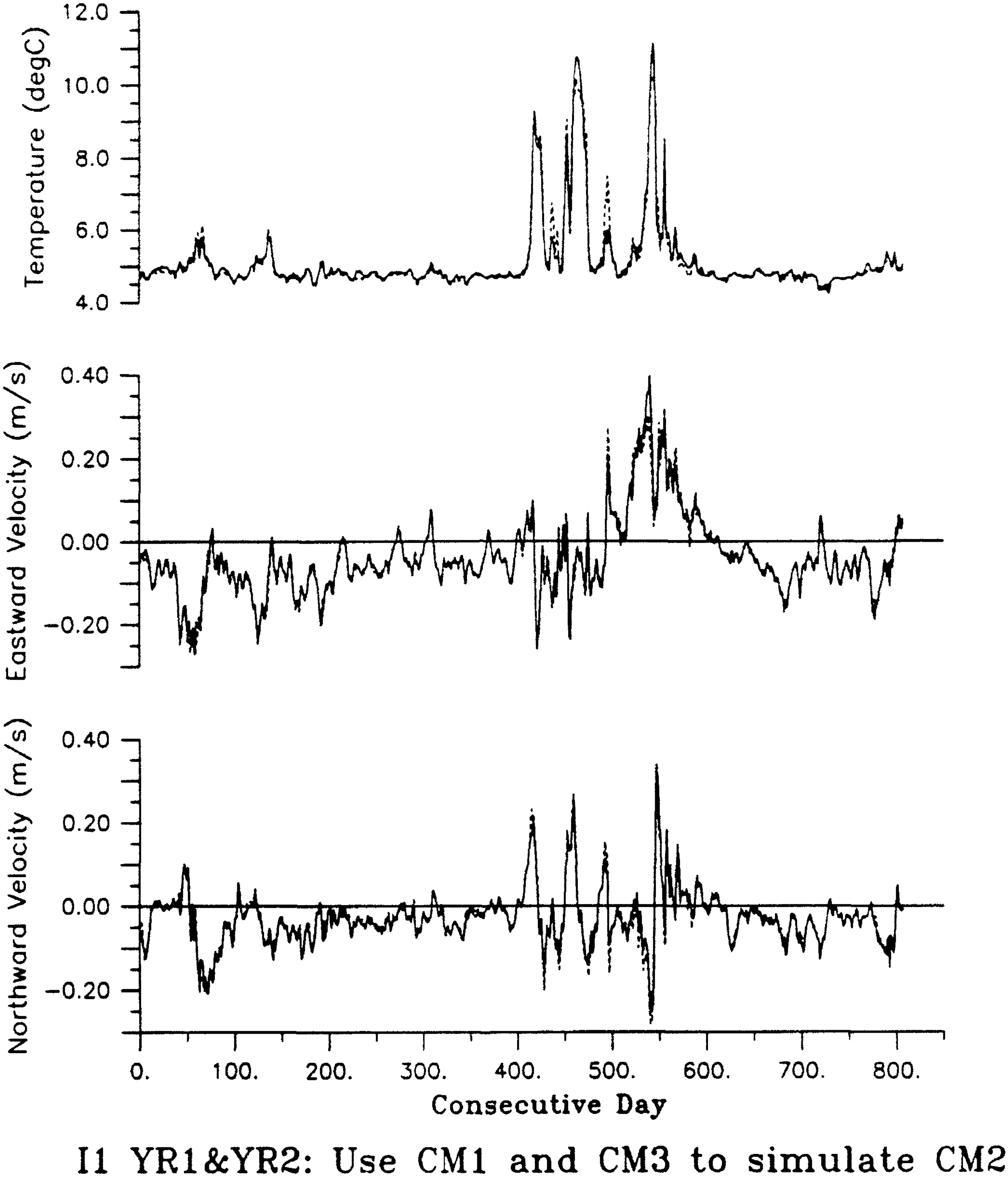


Figure 5a: Results of mooring motion correction Test 1 (interpolation by 300 m) on a northern mooring (I1). Solid lines are direct measurements at level 2; dashed lines are estimated u, v, and T.

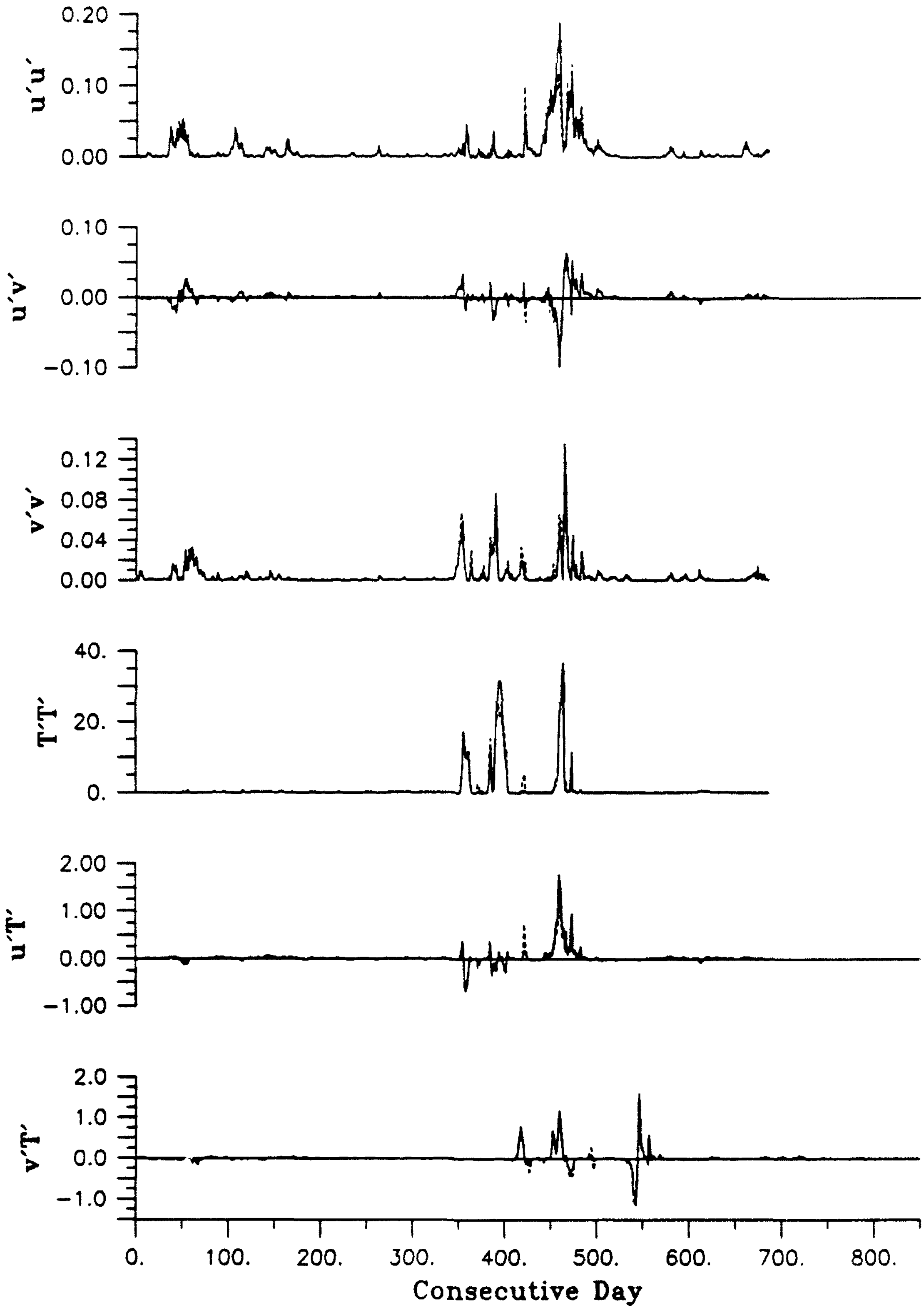


Figure 5b: Same as Figure 5a, except for $u'u'$, $u'v'$, $v'v'$, $T'T'$, $u'T'$, and $v'T'$.

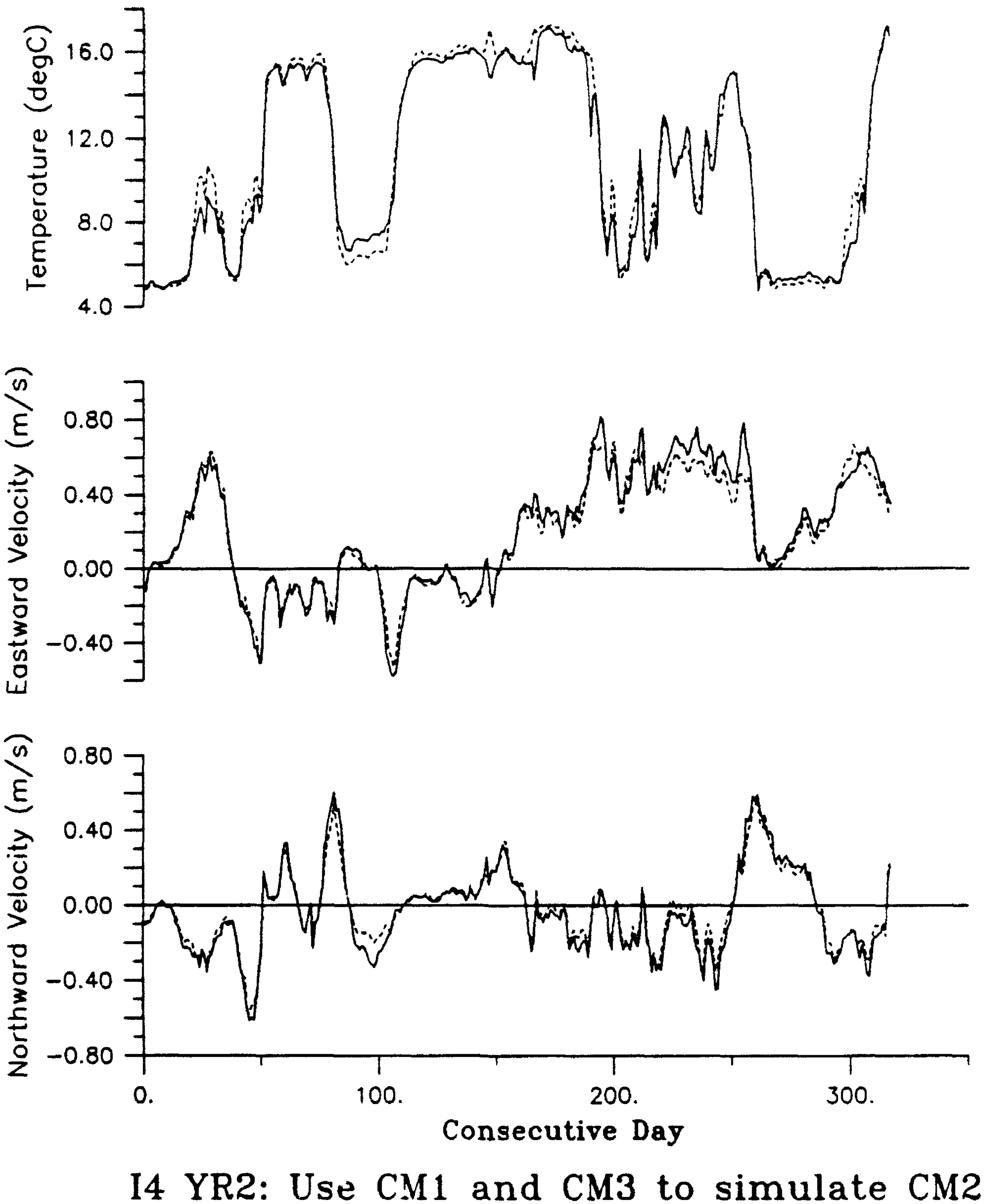


Figure 5c: Results of mooring motion correction Test 1 (interpolation by 300 m) on a mid-stream/southern mooring (I4-YR2). Solid lines are direct measurements at level 2; dashed lines are estimated u, v, and T.

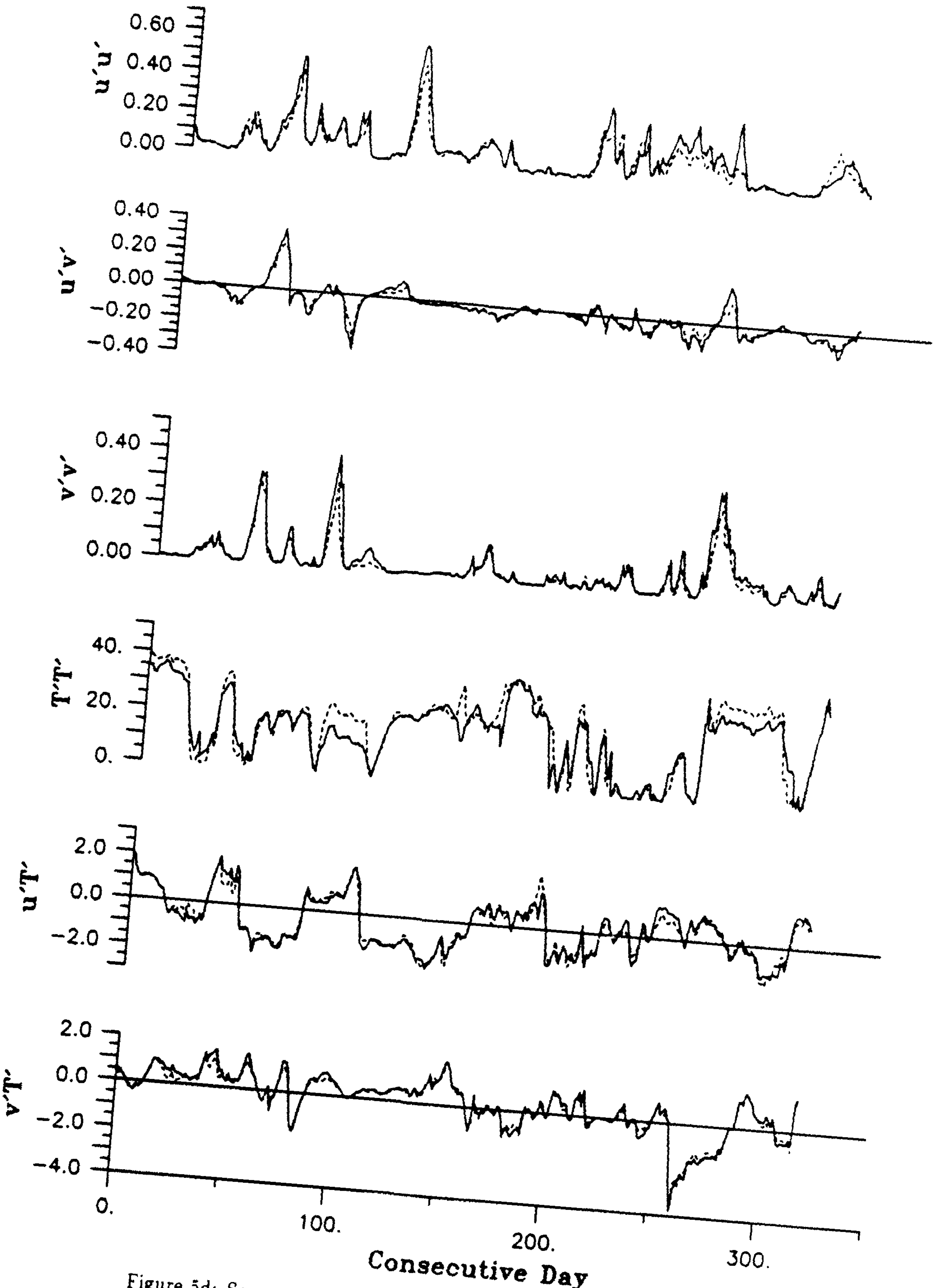


Figure 5d: Same as Figure 5c, except for $u'u'$, $u'v'$, $v'v'$, $T'T'$, $u'T'$, and $v'T'$.

Table 7. Root-mean-square Error Between Measured and Simulated Data for Tests 1 and 2.

Each test was run for both a northern mooring, I1, and a mid-stream/southern mooring, I4_YR2. Test 1 uses current meters at level 1 and 3 to interpolate to level 2. Test 2 uses current meters at level 2 and level 3 to extrapolate to level 1. Units of velocity are m s^{-1}. Temperature units are °C. Figure 4 shows the observed and simulated time series of each test.

	Northern Mooring Site I1		Mid/Southern Mooring Site I4_YR2	
	Test 1	Test 2	Test 1	Test 2
err(T):	0.2	0.6	0.7	0.9
err(u):	0.01	0.08	0.06	0.16
err(v):	0.02	0.07	0.05	0.14
err(u'u'):	0.007	0.068	0.050	0.231
err(u'v'):	0.003	0.034	0.024	0.117
err(v'v'):	0.004	0.040	0.028	0.181
err(T'T'):	1.0	3.7	4.7	6.9
err(u'T'):	0.057	0.340	0.265	0.699
err(v'T'):	0.040	0.286	0.168	0.477

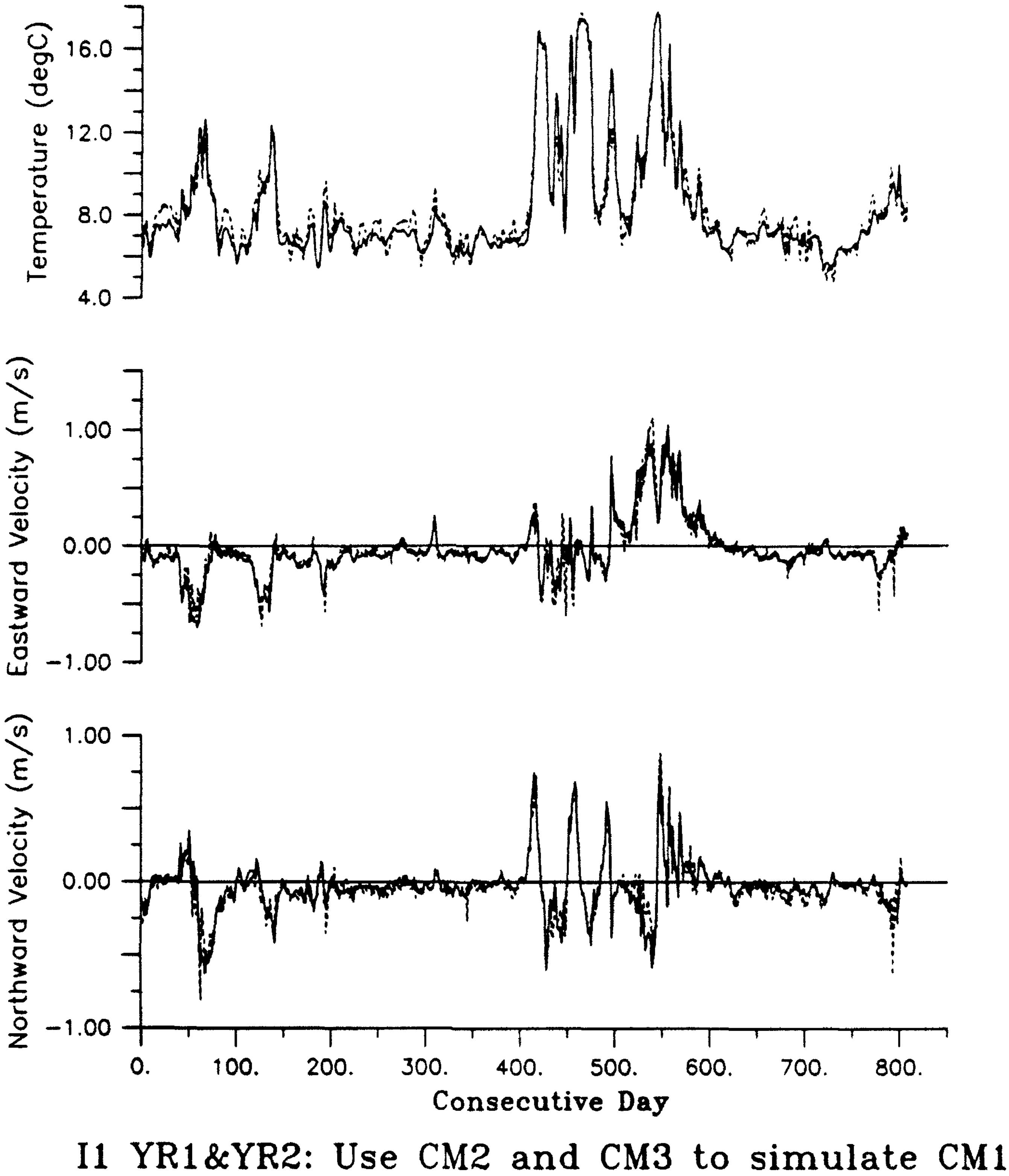


Figure 6a: Results of mooring motion correction Test 2 (extrapolation by 300 m) on a northern mooring (I1). Solid lines are direct measurements at level 1: dashed lines are estimated u, v, and T.

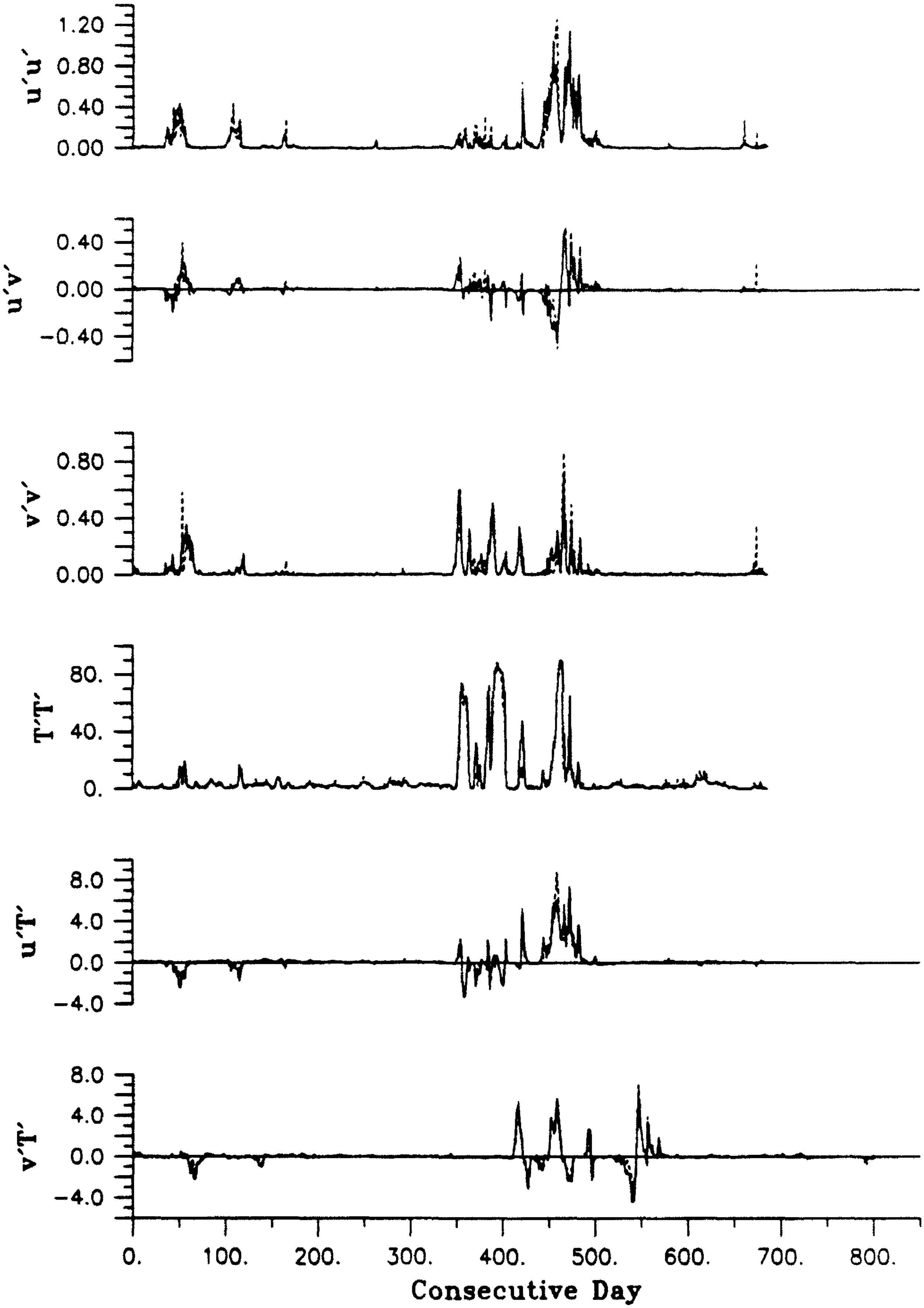


Figure 6b: Same as Figure 6a, except for $u'u'$, $u'v'$, $v'v'$, $T'T'$, $u'T'$, and $v'T'$.

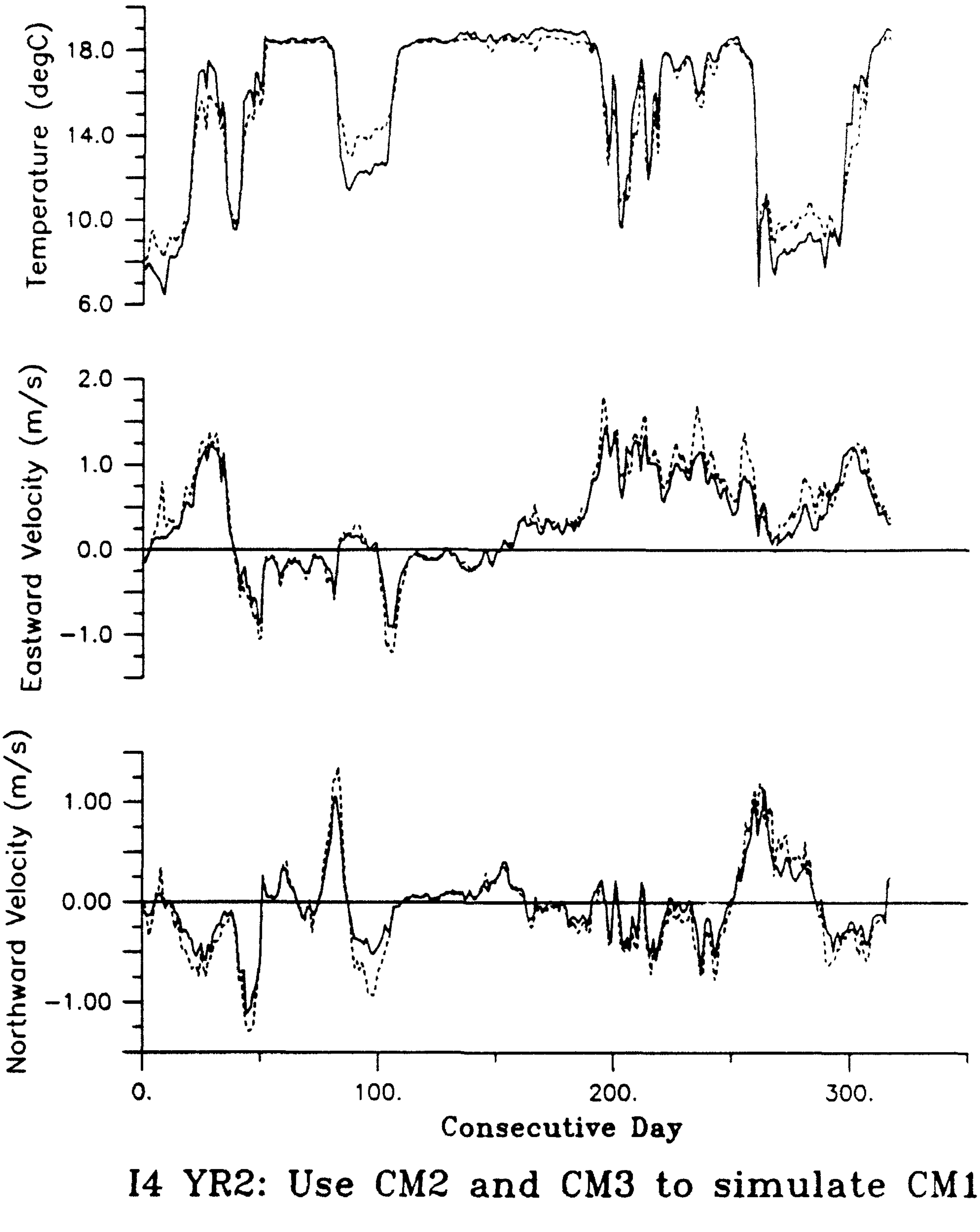


Figure 6c: Results of mooring motion correction Test 2 (extrapolation by 300 m) on a mid-stream/southern mooring (I4_YR2). Solid lines are direct measurements at level 1: dashed lines are estimated u, v, and T.

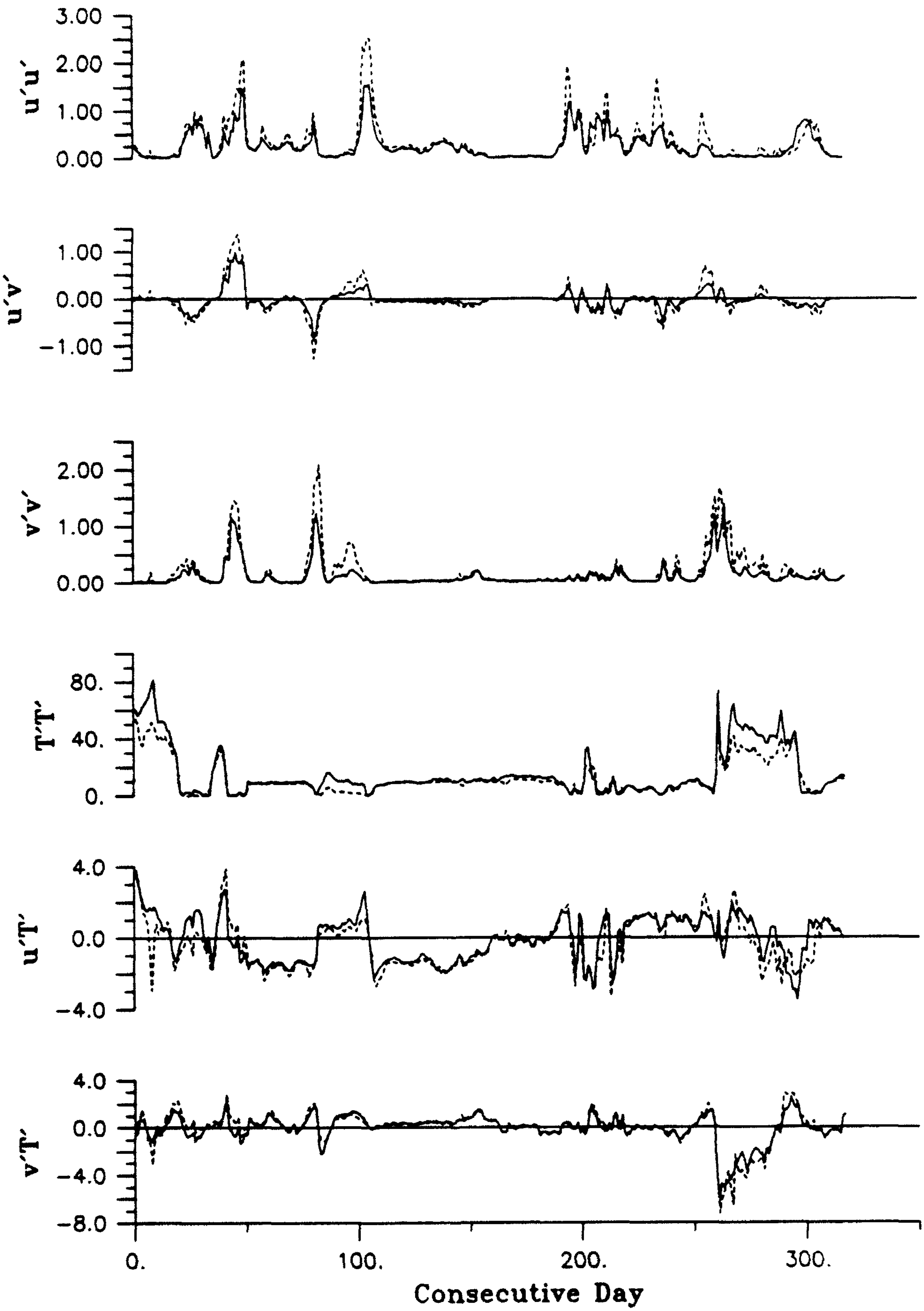


Figure 6d: Same as Figure 6c, except for $u'u'$, $u'v'$, $v'v'$, $T'T'$, $u'T'$, and $v'T'$.

These tests are very strenuous compared to the typical vertical distances used for correcting the temperature and velocity data for mooring motion. Most observed vertical excursions of the current meters required interpolations or extrapolations of less than 50 m to put the data onto the desired pressure horizons.

Our tests show that the method works sufficiently well that data can be simulated at any of the three levels when one of the current meters failed. Figure 3 shows that most of the instrument failures in the Central Array were at level 2. Therefore using the mooring motion correction scheme to interpolate to level 2, the associated errors for those moorings will be similar to the errors presented for Test 1. In addi· on, several of the level 1 current meters had gappy or short velocity records; they include sites G3_YR1, H4_YR2, I3_YR2, and I5_YR2. Thus for those moorings, the velocity and covariance errors at 400 m during the gappy periods will be on the order of those described in Test 2. Because the Hogg (1991) mooring motion correction method works so well, these data gaps, although unfortunate, are not as troublesome as might have been feared.

5 Error Estimations of the Motion Corrected Data

5.1 Estimating the error in T_{cor}

Errors associated with the corrected temperatures arise from both errors in the temperature measurements and errors in the reference pressure. The error in T_{cor} also depends on how the correction was determined, either by using either Equation 3 for one working instrument or Equation 18 for two instruments.

When there is only one working current meter on the mooring, the corrected temperature is $T_{cor} = F(p_{ref} - p_{nom})$. In this case, the correction error is predominantly due to the scatter of the observed temperature-pressure pairs on the canonical profiles, $\sigma_F(p_{ref} - p_{nom})$. However, the measurement errors also cause a small error in the daily reference pressure which in turn affects the corrected temperatures. Thus, the total temperature error is:

$$err(T_{cor}) = err(F) \tag{23}$$

$$= \sqrt{\sigma_F(p_{ref} - p_{nom})^2 + \left(\left.\frac{\partial F}{\partial p_{ref}}\right|_{p_{ref}-p_{nom}} err(p_{ref}) \right)^2} \tag{24}$$

The term $\sigma_F(p_{ref} - p)$ is the standard deviation envelope of observed temperatures around canonical profile. Table 8 lists the observed scatter as a function of pressure for both the northern and southern profiles. The derivative $\frac{\partial F}{\partial p_{ref}}$, can be computed analytically in a straightforward manner because the canonical profiles are modeled as polynomials. However, estimating the error in the reference pressure, $err(p_{ref})$ is not straightforward and is discussed in Section 5.2.

When there are at least two current meters working on the mooring, then the temperature correction can be computed according to Equations 18–22. Accordingly, the error in the corrected temperature is:

$$
\begin{aligned}
err(T_{cor}) \;=\; \Bigg[& (w_u^2 + w_l^2) err(T)^2 \;+\; w_u^2 (\sigma_F(p_{ref} - p_{nom}) - \sigma_F(p_{ref} - p_u))^2 \;+ \\
& w_l^2 (\sigma_F(p_{ref} - p_{nom}) - \sigma_F(p_{ref} - p_l))^2 \;+ \\
& \left(\left.\frac{\partial F}{\partial p_{ref}}\right|_{(p_{ref} - p_{nom})} - w_u \left.\frac{\partial F}{\partial p_{ref}}\right|_{(p_{ref} - p_u)} - w_l \left.\frac{\partial F}{\partial p_{ref}}\right|_{(p_{ref} - p_l)} \right)^2 err(p_{ref})^2 \;+ \\
& \left(\left(\frac{\partial w_u}{\partial p_u} T_u + \frac{\partial w_l}{\partial p_u} T_l \right)^2 + \left(\frac{\partial w_u}{\partial p_l} T_l + \frac{\partial w_l}{\partial p_l} T_l \right)^2 \right) err(p)^2 \Bigg]^{1/2}
\end{aligned}
\tag{25}
$$

The errors of the temperature and pressure observations are $err(T_u) = err(T_l) = err(T) \sim 0.03°C$ and $err(p_u) = err(p_l) = err(p) \sim 5$ db, respectively. The derivative terms can be computed analytically from Equations 2 and 21–22.

Typical values of $err(T_{cor})$ for all three levels are 0.14–0.17°C. The highest errors of 0.27°C were estimated for the 700 m level at site G2_YR2.

5.2 Estimating $err(p_{ref})$

The daily p_{ref} is found by fitting the (T, p) measurements from the working current meters on a mooring to the canonical profile $T = F(p_{ref} - p)$,

$$
0 \;=\; \frac{\partial}{\partial p_{ref}} \left[\sum_{i=1}^{M} (p_{ref} - p_i - F^{-1}(T_i))^2 \right] \tag{26}
$$

Table 8a: RMS Error between the Observed Temperature and the Northern Canonical Profile Temperature

The temperature scatter is listed as a function of $p_{ref} - p$. The units of $p_{ref} - p$ are 1000 kPa. The units of the temperature scatter are °C. The scatter is computed every 20 m using NPTS (T, p) observations. This table was created using Matlab code **tenv.m**, listed in Appendix B.

$p_{ref} - p$	scatter	NPTS
-10.5299	0.2737	3
-10.3299	0.1216	1
-10.1299	0.1730	4
-9.9299	0.1805	8
-9.7299	0.1408	7
-9.5299	0.0927	13
-9.3299	0.0609	20
-9.1299	0.0659	39
-8.9299	0.0528	95
-8.7299	0.0526	132
-8.5299	0.0513	142
-8.3299	0.0469	172
-8.1299	0.0552	132
-7.9299	0.0690	134
-7.7299	0.0565	128
-7.5299	0.0515	133
-7.3299	0.0526	108
-7.1299	0.0625	90
-6.9299	0.0792	67
-6.7299	0.0854	73
-6.5299	0.0895	76
-6.3299	0.0911	62
-6.1299	0.1141	60
-5.9299	0.0883	86
-5.7299	0.0693	129
-5.5299	0.0879	164
-5.3299	0.0800	167
-5.1299	0.0719	194
-4.9299	0.0713	154
-4.7299	0.0974	143
-4.5299	0.1181	150
-4.3299	0.1078	132
-4.1299	0.1488	126
-3.9299	0.1427	102
-3.7299	0.1448	76
-3.5299	0.1582	75
-3.3299	0.1812	83
-3.1299	0.2466	73

$p_{ref} - p$	scatter	NPTS
-2.9299	0.2268	71
-2.7299	0.1914	110
-2.5299	0.1668	157
-2.3299	0.1960	160
-2.1299	0.2029	190
-1.9299	0.1420	129
-1.7299	0.1212	132
-1.5299	0.1673	138
-1.3299	0.1355	145
-1.1299	0.1222	115
-0.9299	0.0855	107
-0.7299	0.0726	72
-0.5299	0.0670	79
-0.3299	0.0807	73
-0.1299	0.0792	63
0.0701	0.1282	64
0.2701	0.1459	47
0.4701	0.1364	46
0.6701	0.1907	42
0.8701	0.2519	45
1.0701	0.2350	43
1.2701	0.2576	23
1.4701	0.3842	21
1.6701	0.2793	29
1.8701	0.3329	20
2.0701	0.2490	34
2.2701	0.1690	29
2.4701	0.1359	28
2.6701	0.1391	16
2.8701	0.2021	11
3.0701	0.2497	13
3.2701	0.2273	13
3.4701	0.2089	8
3.6701	0.1109	9
3.8701	0.0778	8
4.0701	0.0635	14
4.2701	0.1281	3

Table 8b: RMS Error between the Observed Temperature and the Southern Canonical Profile Temperature

The temperature scatter is listed as a function of $p_{ref} - p$. The units of $p_{ref} - p$ are 1000 kPa. The units of the temperature scatter are °C. The scatter is computed every 20 m using NPTS (T, p) observations. This table was created using Matlab code **tenv.m**, listed in Appendix B.

$p_{ref} - p$	scatter	NPTS
−9.7271	0.0957	4
−9.5271	0.0856	2
−9.3271	0.0555	6
−9.1271	0.0740	16
−8.9271	0.1429	15
−8.7271	0.1104	34
−8.5271	0.1077	46
−8.3271	0.1038	59
−8.1271	0.0931	59
−7.9271	0.0960	78
−7.7271	0.0735	76
−7.5271	0.0785	49
−7.3271	0.0825	51
−7.1271	0.1025	31
−6.9271	0.0924	31
−6.7271	0.0883	37
−6.5271	0.1069	32
−6.3271	0.1195	31
−6.1271	0.1268	47
−5.9271	0.1288	53
−5.7271	0.1328	47
−5.5271	0.1088	68
−5.3271	0.1028	95
−5.1271	0.1087	129
−4.9271	0.1254	179
−4.7271	0.1258	139
−4.5271	0.1802	132
−4.3271	0.2028	99
−4.1271	0.2171	103
−3.9271	0.2054	76
−3.7271	0.2146	79
−3.5271	0.2568	81
−3.3271	0.2875	87
−3.1271	0.3615	95
−2.9271	0.3240	105
−2.7271	0.3546	102
−2.5271	0.4385	112
−2.3271	0.3276	147
−2.1271	0.3019	212
−1.9271	0.3691	256
−1.7271	0.3309	328
−1.5271	0.2575	348
−1.3271	0.3864	254
−1.1271	0.3581	159
−0.9271	0.2965	142
−0.7271	0.1896	122
−0.5271	0.1184	83
−0.3271	0.1200	82
−0.1271	0.1327	74
0.0729	0.1809	93
0.2729	0.2083	79
0.4729	0.2465	83
0.6729	0.3151	71
0.8729	0.2905	100
1.0729	0.3616	124
1.2729	0.4376	141
1.4729	0.3736	193
1.6729	0.3330	234
1.8729	0.6147	150
2.0729	0.6546	86
2.2729	0.5602	56
2.4729	0.4221	56
2.6729	0.4057	44
2.8729	0.2985	56
3.0729	0.3170	51
3.2729	0.2745	51
3.4729	0.2316	55
3.6729	0.2940	46
3.8729	0.2476	62
4.0729	0.2165	97
4.2729	0.2094	103
4.4729	0.1817	142
4.6729	0.1859	206
4.8729	0.1786	162
5.0729	0.2318	79
5.2729	0.2207	67
5.4729	0.1970	40
5.6729	0.2276	16
5.8729	0.2070	5
5.8729	0.2070	5
6.2729	0.5220	1

$$p_{ref} = \sum_{i=1}^{M} \frac{p_i - F^{-1}(T_i)}{M} \tag{27}$$

where M is the number of current meters on the mooring. Thus, assuming each of the M instruments have random temperature and pressure measurement errors, the error in the reference pressure is

$$err(p_{ref}) = \sqrt{M\left(\frac{err(p)}{M}\right)^2 + \sum_{i=1}^{M}\left(\frac{\partial}{\partial T}F^{-1}(T_i)\frac{err(T)}{M}\right)^2} \tag{28}$$

$$= \sqrt{\frac{err(p)^2}{M} + \left(\frac{err(T)}{M}\right)^2 \sum_{i=1}^{M}\left(\frac{\partial}{\partial T}F^{-1}(T_i)\right)^2} \tag{29}$$

Rather than inverting the polynomial, $F(p_{ref} - p)$, we found it simpler to fit the northern data and southern data to an arctanh function, $G(T)$:

$$p_{ref} - p = G(T) \tag{30}$$

$$= C_2 \operatorname{atanh}\left(\frac{T - C_1}{C_3}\right) + C_4 \tag{31}$$

such that $G(T) \sim F^{-1}(T)$. Thus,

$$\frac{\partial F^{-1}(T)}{\partial T} \sim \frac{\partial G(T)}{\partial T} = \frac{C_2}{C_3}\left(\frac{1}{1 - arg^2}\right) \tag{32}$$

$$\text{where,} \qquad arg = \frac{T - C_1}{C_3} \tag{33}$$

Note that this derivative is defined only for $-1 < arg < 1$. Because the observed temperatures were sometimes different than the canonical profile, values of arg that were less than or equal to -1, were replaced by -0.99. Likewise, values of arg greater than or equal to 1 were replaced by $+0.99$.

Values of $err(p_{ref})$ for all three levels ranged between 0.042 kPa and 0.10 kPa. The mean value for all sites was 0.061 kPa.

5.3 Estimating the error in U_{cor}

The error in the corrected velocity (U_{cor}) depends on (i) the measurement error $err(U) \sim 2$ cm s^{-1}, (ii) the error in the corrected temperature, $err(T_{cor})$, which was discussed above, (iii) the error in the angle of the shear, $err(\theta) \sim 5°$, and (iv) the error in the assumption that the

change in the velocity shear is proportional to the change in the temperature, $err(m)$. The error in m depends upon whether the velocity correction is an interpolation or an extrapolation. For interpolation where $T_u \geq T_{cor} \geq T_l$, we found $err(m) \sim 0.01$. Otherwise, $err(m) \sim 0.02$.

Assuming these errors are independent,

$$err(\mathbf{U_{cor}}) = \sqrt{\left(\frac{\partial U_{cor}}{\partial \theta} err(\theta)\right)^2 + \left(\frac{\partial U_{cor}}{\partial T_{cor}} err(T_{cor})\right)^2 + \left(\frac{\partial U_{cor}}{\partial m} err(m)\right)^2 + err(U)^2} \quad (34)$$

The partial derivatives can be determined by examining Equation 14.

Mean $err(\mathbf{U_{cor}})$ values of 0.02 m s^{-1} were obtained for all moorings except three (sites G3_YR1, I3_YR1, and G3_YR2) where the mean errors were 0.15–0.17 m s^{-1}.

6 Useful By-products of the Correction Scheme

The canonical profile can be exploited in a variety of ways to obtain additional data products.

6.1 The Pseudo-IES

Because the canonical profile is represented by a Nth order polynomial whose zeroth order coefficient is set to be 12°C, the reference pressure is the pressure of the 12°C isotherm, $T(p = p_{ref}) = F(0) = 12$°C. Furthermore, the reference pressure can be divided by a factor of 1.01 to convert pressure into depth in meters. Thus we can obtain a time series of the depth of the 12°C isotherm at the current meter site.

This is the same measurement obtained by the IESs which were also deployed in the Central Array. As discussed in Section 2.3, we used the IES Z_{12} records as p_{ref} for two sites when several current meters on those moorings failed. Likewise, the moorings can be used as "pseudo-IES"s to help map the thermocline topography where the IESs failed. Appendix D show comparisons of the Z_{12} measurements from IESs and pseudo-IESs at all the mooring sites. Typically, the rms differences between the two time series are under 25 m, which is less than the error of the objectively mapped IES Z12.

6.2 Computing the Mean Stratification

Because the SYNOP moorings were arranged along lines approximately perpendicular to the mean Gulf Stream path, the mean temperature profiles at each mooring along a line can be contoured into a Eulerian mean temperature cross-section. Figure 7 shows both the year 1 and year 2 mean temperature cross-sections along the I line (near 68°W).

Furthermore, because the canonical profile has an analytical form, the Brunt-Vaisalla frequency, $\overline{N^2} = \overline{g\alpha\frac{\partial T}{\partial p}}$ (where $\alpha = -\frac{1}{\rho_0}\frac{d\rho}{dT}$ and the overline indicates a time average) can be computed simply as:

$$\overline{N^2} = \overline{g\alpha \sum_{n=1}^{N-1} (N+1-n)\, c_n\, (p_{ref} - p)^{(N-n)}}. \tag{35}$$

The year 1 and year 2 mean stratification cross-sections along the I line are shown in Figure 8.

7 Summary

Hogg (1991) provides a robust mooring motion correction method. We made a slight modification to his method however, by using a weighted average of the corrections of the two nearest temperature measurements. This revision allows the corrected temperature to equal the measured temperature when current meter is at the nominal pressure.

The tests discussed in Section 4 show that even when there were only two working current meters on a mooring, the temperature and velocity measurements could be 'corrected' to all three nominal levels.

A feature of this method is that it uses the uncorrected data to create the canonical profiles and reference pressure. Thus, this scheme provides a method of correcting temperature and velocity data for mooring motion, even in the absence of historical data sets.

Because of the wealth of data in the SYNOP Central Array, other types of measurements could be incorporated into the correction scheme to improve the velocity and temperature corrections. Specifically, ADCP pressure measurements were used whenever possible, because they could be verified by independent acoustic calculations of the depth. Additionally, the ADCP temperatures and Bin 1 velocities were used whenever possible to fill data gaps left by the current meters. Furthermore, IES Z_{12} data were used as the reference pressures for

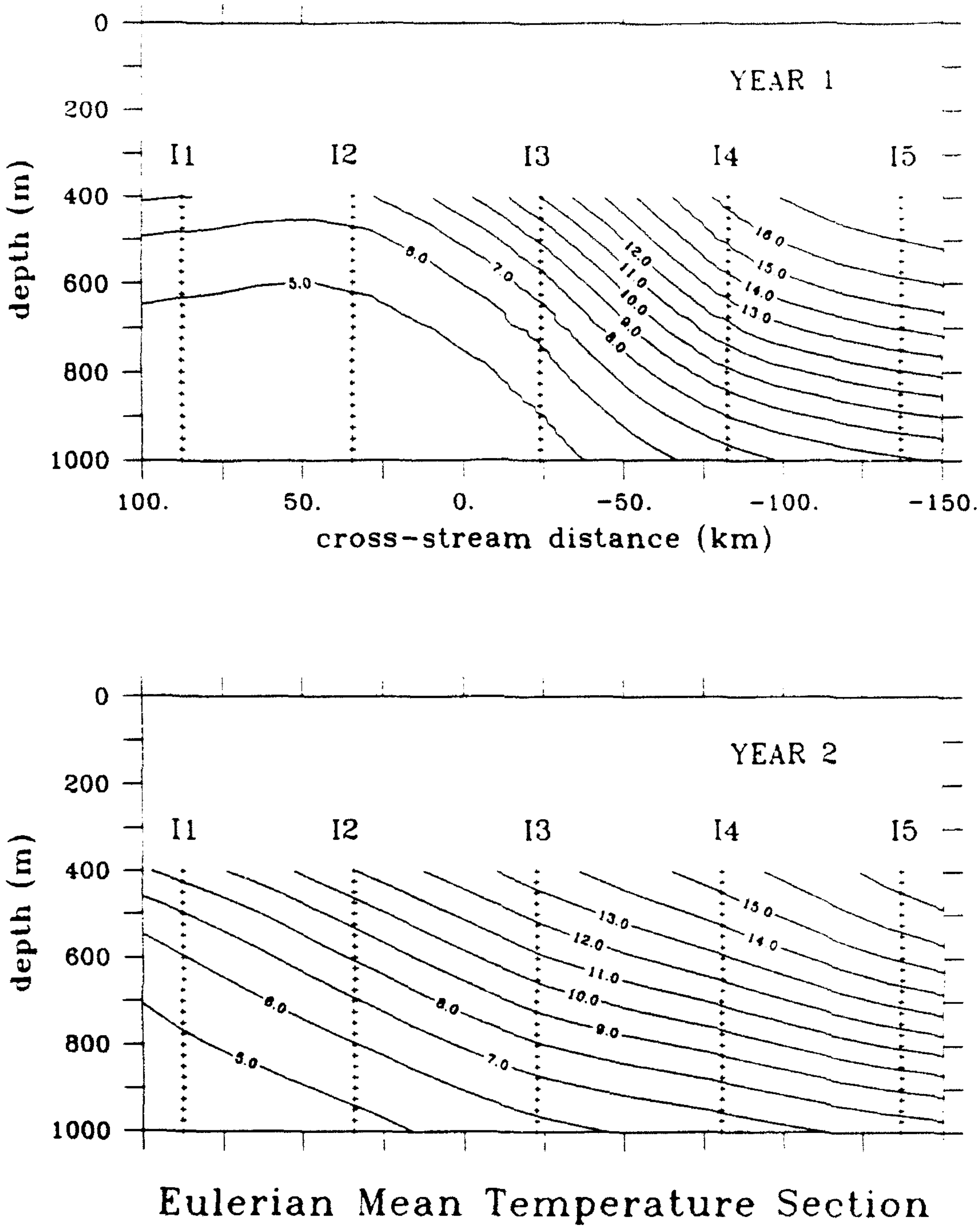


Figure 7: The Year 1 (June 1988–June 1989) and Year 2 (August 1989–August 1990) Eulerian mean temperature cross-sections measured by the moorings along Line I.

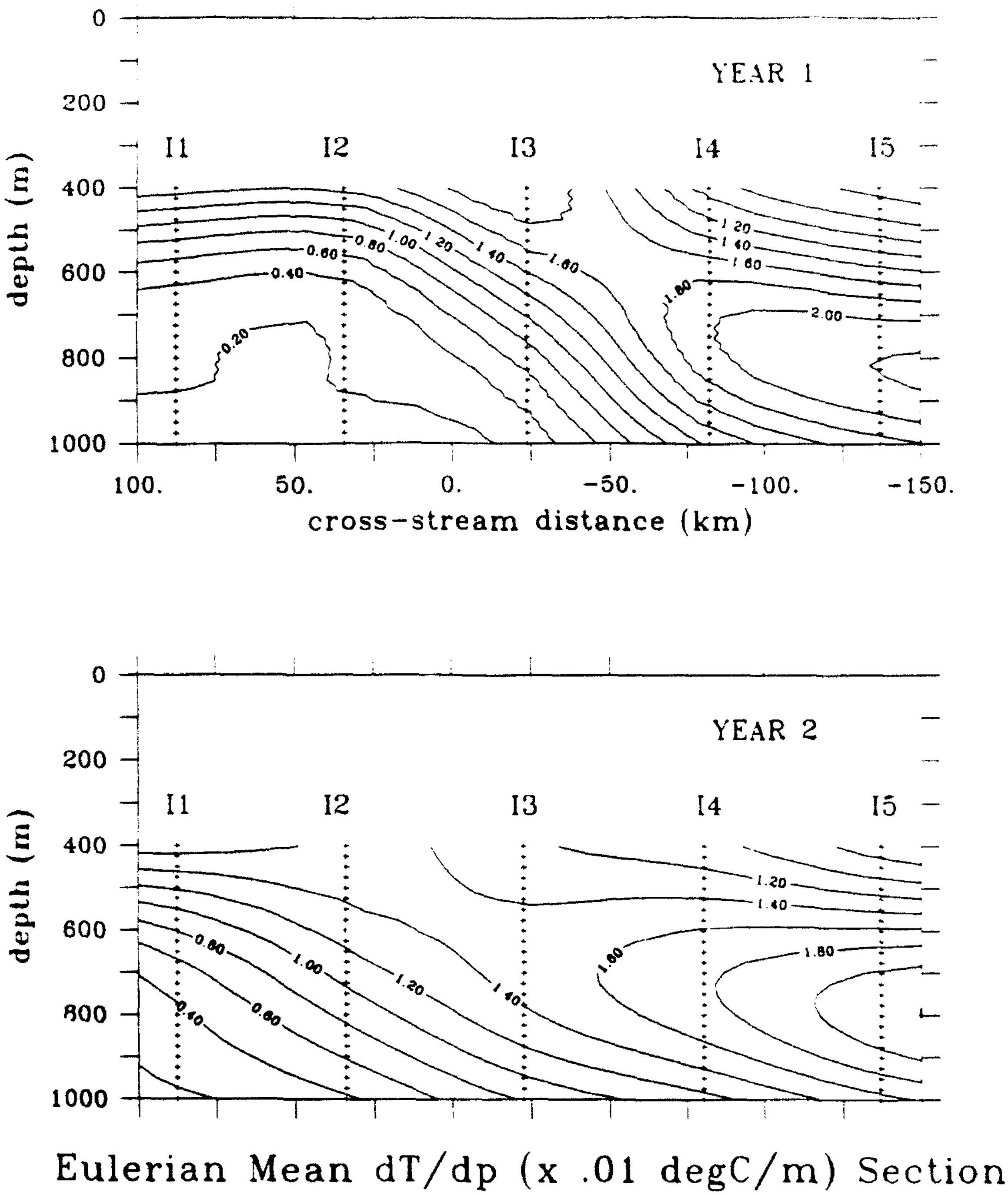


Figure 8: The Year 1 (June 1988–June 1989) and Year 2 (August 1989–August 1990) Eulerian mean stratification ($\frac{\partial \overline{T}}{\partial z}$) measured by the moorings along Line I.

moorings that had only one of the upper level current meters functioning properly.

Finally, we have also noted that the moorings can be used as pseudo-IESs. We have already done this by incorporating their p_{ref} records into our objective maps of the thermocline field. The inclusion of these current meter data into the IES maps sharpened the thermocline gradients and improved the maps in regions where there were no IES measurements.

Altogther, Hogg's (1991) mooring motion correction scheme allows the IES and current meter data to be mutually beneficial.

Acknowledgements

We thank Nelson Hogg for generously sharing with us his mooring motion correction Matlab code.

The SYNOP Experiment was supported by the Office of Naval Research under contract numbers N00014-90J-1568 and N00014-90J-1548 and the National Science Foundation under grant number OCE97-17144.

References

Hogg, N. G., 1991. Mooring Motion Corrections Revisited. *J. Atmos. Oceanic Technol.*, 8, 289-295.

Shay, T.J., S. Haines, J. M. Bane, and D. R. Watts, 1993. SYNOP Central Array current meter data report:Mooring period May 1988–September 1990. Univ. North Carolina Technical Report.

Tracey, K. L. and D. R. Watts, 1991. The SYNOP Experiment, Thermocline depth maps for the Central Array, October 1987 to August 1990. Univ. Rhode Island, GSO Technical Report 91-5. 193 pp.

Appendix A: ADCP Temperature Evaluations

Due to differences in the calibration procedures, we believe the absolute temperatures of the level 1 current meter ($T1$) but not those of the ADCP (T_{ADCP}). However, we know that in general T_{ADCP} should be approximately $0.2 - 0.25°C$ warmer than $T1$ since the ADCP was located 12 m above the current meter. This is based on a typical thermocline gradient of 1°C for $48 - 60$ m. Since these sites were located in the northern portion of the array where the thermocline is frequently shallow, the ADCPs (at roughly 400 m depth) were sometimes below the thermocline in colder water. In those cases, we might expect a smaller thermal gradient, e.g. $0.15°C/12$ m $= 1°C/80$ m, corresponding to temperature differences of 0.15°C. Likewise in 18°C water, we might expect a *very* small gradient. However, except for rare intrusion events, the gradients should always be positive since the ADCPs are positioned above the current meters on the moorings.

Comparisons of the $T1$ and T_{ADCP} records were made for all four of ADCPs. These included the full records for sites H3_YR1, H3_YR2 and H4_YR1, as well as a short record for site I2_YR2, when the current meter failed after a 4-month period. No ADCP data were obtained for sites I2_YR1 and H4_YR2. The results of these comparisons are summarized here. The error in these temperature offsets is $\pm - 0.05°C$. "Good" means that the difference $\Delta_T = T_{ADCP} - T1$ corresponds to a realistic temperature gradient. Plots of the temperature difference Δ_T versus $T1$ are shown in Figure 9.

H3_YR1:

T_{ADCP} appears to have an offset of approximately $-0.5°C$. It's too cold.

- The ADCP temperature is *always colder* than the current meter $T1$. Thus T_{ADCP} is *bad.*
- For $T1 = 6 - 10°C$, $\Delta_T = -0.4$ to $-0.25°C$. It should be $+0.15$ to $+0.2°C$.
- For $T1 = 10 - 14°C$, $\Delta_T = -0.3$ to $-0.1°C$. It should be $+0.25°C$.
- $T_{ADCP} - T1$ is less than $-0.5°C$ during one *cold* event ($T1 = 6°C$). This is probably an intrusion.

H4_YR1:

- T_{ADCP} looks like it has an offset of -0.25 to $-0.3°C$. It is *too cold.*

- For $T1 = 5 - 10°C$, $\Delta_T = -0.15$ to $-0.10°C$. It should be -0.15 to $-0.2°C$.
- For $T1 = 10 - 17°C$, $\Delta_T = -0.05$ to $+0.25°C$. It should be around $0.25°C$.
- For $T1 = 18°C$, $\Delta_T = -0.20$ to $-0.25°C$. It should be $0.0 - 0.1°C$.

H3_YR2:

- T_{ADCP} looks good. If anything, there might be an offset of $-0.05°C$, which is too cold.
- For $T1 = 7 - 10°C$, $\Delta_T = 0.05$ to $0.2°C$.
- For $T1 = 10 - 16°C$, $\Delta_T = 0.2$ to $0.4°C$.
- For $T1 = 18°C$, $\Delta_T = +0.0$ to $0.05°C$
- There are 2–3 spikes when T_{ADCP} was about 0.02°C or so colder than $T1$. Overall however, this is a very reasonable record.

I2_YR2:

- T_{ADCP} looks good. If anything, the T_{ADCP} might be 0.1° C too warm.
- For $T1 = 6 - 10°$ C, $= 0.2°C$.
- For $T1 = 10 - 16°C$, $\Delta_T = 0.25 - 0.5°C$, and occasionally $\Delta_T = 0.5 - 0.7$ C warmer.
- The T_{ADCP} is never colder than $T1$ over the 500-point record (4 months).

In summary, these comparisons show that it is reasonable to use the ADCP temperatures in the mooring motion correction of sites H3_YR2 and I2_YR2. However, extra precautions should be taken when using the ADCP temperatures of sites H3_YR1 and H4_YR1. We did not use either of these two records for the mooring motion corrections described in this report.

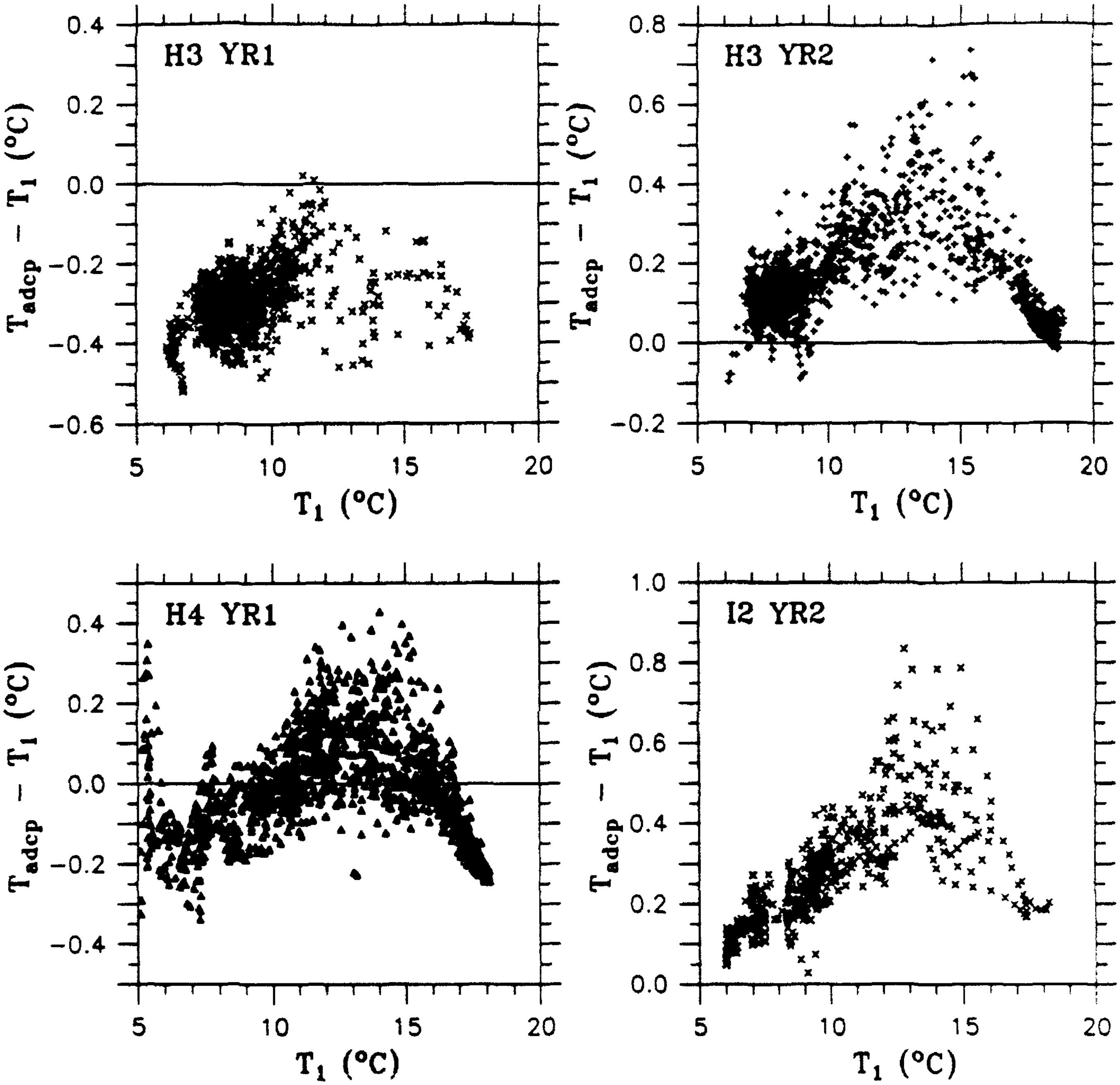


Figure 9: The temperature differences $\Delta T = T_{ADCP} - T_1$ are plotted as a function of the level 1 current meter temperature T_1.

Appendix B: Mooring Motion Correction Matlab Codes

There are two basic driver programs associated with the mooring motion correction scheme. The first computes the canonical profile from a subset of the temperature and pressure data, and the second uses the canonical profile to correct the temperature and velocity data of a current meter to the nominal depth. However because we classified the moorings as either northern and southern sites, we needed two distinct canonical profiles. Thus we created two versions of each of the driver programs, one version for the northern sites and the other for the southern sites. For the northern sites, the program **synopbznorth.m** computes the canonical profile and **mmcorn.m** uses that profile to correct the temperature and velocity data of a single current meter on a given mooring. The programs **synopbzmid.m** and **mmcors.m** are the respective codes for the southern moorings.

The driver programs for determining the canonical profiles, **synopbznorth.m** and **synopbzmid.m**, each call three subroutines: (1) **mcor.m** performs the iteration to determine the coefficients of the polynomial. (2) **nlreg.m** iterates to determine the reference pressure. (3) **errbzp.m** calculates the polynomial and its derivative for each day.

The other two driver programs, **mmcorn.m** and **mmcors.m**, each call four subroutines: (1) **zest.m** computes the daily reference pressure. (2) **tempcor.m** computes the corrected temperature and estimates the associated error. (3) **tprefer.m** computes the error in the reference pressure, and (4) **velc.m** computes the corrected velocity and estimates its error.

The error estimates depend on knowing the standard deviation envelope of the temperature measurements about the canonical profile. For completeness, the code, **tenv.m**, which computes the lookup table for the standard deviation envelope, is included in this appendix.

```
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%
%                              SYNOPBZNORTH.M
%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% synopbznorth.m          % Compute polynomial coeficents of the
%                           Northern Canonical profile.
%
% input:  t - temp data from northern moorings used in regression
%                note: 0 is the flag for no data
%
% output: cn - coeficients of canonical profile
%         zsyn - pref + pl for input t
%
% subroutines:  mcor m    (which calls: nlreg.m and errbzp.m)
%               nlreg.m
%               errbzp.m
%
% save session in north7.mat
%

!cat /usr/users/cm/40hlp/newfiles/h2_1.year2.40hlp | premat > temph21.dat
!cat /usr/users/cm/40hlp/newfiles/h2_2.year2.40hlp | premat > temph22.dat
!cat /usr/users/cm/40hlp/newfiles/h2_3.year2.40hlp | premat > temph23.dat

!cat /usr/users/cm/40hlp/newfiles/i1_1.year2.40hlp | premat > tempi11.dat
!cat /usr/users/cm/40hlp/newfiles/i1_2.year2.40hlp | premat > tempi12.dat
!cat /usr/users/cm/40hlp/newfiles/i1_3.year2.40hlp | premat > tempi13.dat

!cat /usr/users/cm/40hlp/newfiles/g2_1.year1.40hlp | premat > templg21.dat
!cat /usr/users/cm/40hlp/newfiles/g2_2.year1.40hlp | premat > templg22.dat

!cat /usr/users/cm/40hlp/newfiles/h3_1.year1.40hlp | premat > templh31.dat
!cat /usr/users/cm/40hlp/newfiles/h3_2.year1.40hlp | premat > templh32.dat
!cat /usr/users/cm/40hlp/newfiles/h3_3.year1.40hlp | premat > templh33.dat

!cat /usr/users/cm/40hlp/newfiles/h3_1.year2.40hlp | premat > temp2h31.dat
!cat /usr/users/cm/40hlp/newfiles/h3_2.year2.40hlp | premat > temp2h32.dat
!cat /usr/users/cm/40hlp/newfiles/h3_3.year2.40hlp | premat > temp2h33.dat

!cat /usr/users/cm/40hlp/newfiles/i2_2.year2.40hlp | premat > temp2i22.dat
!cat /usr/users/cm/40hlp/newfiles/i2_3.year2.40hlp | premat > temp2i23.dat

!cat /usr/users/cm/40hlp/newfiles/h4_1.year2.40hlp | premat > temp2h41.dat
!cat /usr/users/cm/40hlp/newfiles/h4_2.year2.40hlp | premat > temp2h42.dat
!cat /usr/users/cm/40hlp/newfiles/h4_3.year2.40hlp | premat > temp2h43.dat

!cat /usr/users/cm/40hlp/newfiles/i3_1.year1.40hlp | premat > templi31.dat
!cat /usr/users/cm/40hlp/newfiles/i3_3.year1.40hlp | premat > templi33.dat

load temph21.dat
load temph22.dat
load temph23.dat

load tempi11.dat
load tempi12.dat
load tempi13.dat

load templh31.dat
load templh32.dat
load templh33.dat

load temp2h31.dat
```



```
load temp2h32.dat
load temp2h33.dat

load temp2i22.dat
load temp2i23.dat

load templg21.dat                    %top instrument u,v,t,p
load templg22.dat

load temp2h41.dat
load temp2h42.dat
load temp2h43.dat

load templi31.dat
load templi33.dat

      th21 - temph21(:,2);
      th22 - temph22(:,2);
      th23 - temph23(:,2);

      ti11 - tempi11(:,2);
      ti12 - tempi12(:,2);
      ti13 - tempi13(:,2);

      t1h31 - templh31(:,2);
      t1h32 - templh32(:,2);
      t1h33 - templh33(:,2);

      t2h31 - temp2h31(:,2);
      t2h32 - temp2h32(:,2);
      t2h33 - temp2h33(:,2);

      t2i22 - temp2i22(:,2);
      t2i23 - temp2i23(:,2);

      t1g21 - templg21(:,2);
      t1g22 - templg22(:,2);

      t2h41 - temp2h41(:,2);
      t2h42 - temp2h42(:,2);
      t2h43 - temp2h43(:,2);

      t1i31 - templi31(:,2);
      t1i33 - templi33(:,2);

% Want all CMs on mooring to start and end at same time

      nh2 - 366:length(th22); % delta p   308  2.04
      ni1 - 366:length(ti12); % delta p   309  2.04
      n1h3 - 1:length(t1h33); % delta p   306  2.04
      n1g2 - (1:length(t1g22))';  % delta p - 304 2.03
      n1i3 - (1:length(t1i33))',

% Load temperature array: [topCM(p) midCM(p 3.08) botCM(p 6.16)],

 t -  [th21(nh2)       th22(nh2)  th23(nh2),     ...   % 3.08  2.04
       ti11(ni1)        ti12(ni1)  ti13(ni1),          % 3.09  2.04
        zeros(length(t2i22),1)  t2i22  t2i23,    ...   % 3.06  2.04
         t1h31(n1h3)  t1h32(n1h3)  t1h33(n1h3)],       % 3.06  2.04
 t - [t;  t2h31           t2h32         t2h33,         % 3.06  2.03
          t1i31(n1i3)   zeros(n1i3) t1i33],            % 3.06  2.03

%          t2h41          t2h42          t2h43,        % 3.04  2.03
%          t1g21(n1g2)    t1g22          zeros(n1g2)], % 3.04  2.03
```

```
nlen = 1:length(t);   %useful for plot(nlen,t(:,1),nlen,t(:,2),nlen,t(:,3))

%
%set up parameters for regression, 3levels
%
%  Will fit 2-3 levels (p) of temp data to polynomial:
%  t(p) = cn(1)*(z+pl-p)^nc + cn(2)*(z+pl-p)^(nc-1) +...+ cn(nc)*(z+pl-p) + 12;
%
%  THUS, t(p=pref=z+pl) = 12 deg C  i.e.  pref = psuedo Z12 !!!!
%
%  where note: delp = -(pl - p(input data)) = [0 3.08 6.16];
%

nc=7,    %order of polynomial
err=.002;
delp=[0 3.07 6.25]; %pressure offsets from top instrument
global z
global delp
np = length(t),

if(isempty(z))        %initialize regression coeficients
        z=6+zeros(1,np);
        c0=[zeros(1,nc-1),1],%initial guess at coeficients for nc order
end
[cn,it]=mcor(c0,t,err,10,1)     % include 5th argument if want fancy regression

zp=min(z):max(delp):.1:max(z);
tp=polyval([cn 0],zp)+12;

plot(t(:,1),z,'w.',t(:,2),z-delp(2),'g.',t(:,3),z-delp(3),'b.',tp,zp,'+')

zsyn = z; %save so that can redo profile plot. z is used again in driver.
!rm temp*.dat   %clean up directory.
```

```
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%
%                          SYNOPBZMID.M
%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% synopbzmid.m          % Compute polynomial coeficents of the
%                         Southern Canonical profile.
%
% input:  t - temp data from southern moorings used in regression
%               note: 0 is the flag for no data
%
% output: cn = coeficients of canonical profile
%         zsyn = pref + pl for input t
%
% subroutines:  mcor.m    (which calls: nlreg.m and errbzp.m)
%               nlreg.m
%               errbzp.m
%
% save session in mid9.mat
%

!cat /usr/users2/cm/40hlp/newfiles/g3_1.year1.40hlp | premat > templg31.dat
!cat /usr/users2/cm/40hlp/newfiles/g3_2.year1.40hlp | premat > templg32.dat
!cat /usr/users2/cm/40hlp/newfiles/g3_3.year1.40hlp | premat > templg33.dat

!cat /usr/users2/cm/40hlp/newfiles/h3_1.year1.40hlp | premat > templh31.dat
!cat /usr/users2/cm/40hlp/newfiles/h3_2.year1.40hlp | premat > templh32.dat
!cat /usr/users2/cm/40hlp/newfiles/h3_3.year1.40hlp | premat > templh33.dat

!cat /usr/users2/cm/40hlp/newfiles/h3_1.year2.40hlp | premat > temp2h31.dat
!cat /usr/users2/cm/40hlp/newfiles/h3_2.year2.40hlp | premat > temp2h32.dat
!cat /usr/users2/cm/40hlp/newfiles/h3_3.year2.40hlp | premat > temp2h33.dat

!cat /usr/users2/cm/40hlp/newfiles/h6_1.year2.40hlp | premat > temp2h61.dat
!cat /usr/users2/cm/40hlp/newfiles/h6_3.year2.40hlp | premat > temp2h63.dat

!cat /usr/users2/cm/40hlp/newfiles/ml3_1.year2.40hlp | premat > tempml31.dat
!cat /usr/users2/cm/40hlp/newfiles/ml3_2.year2.40hlp | premat > tempml32.dat
!cat /usr/users2/cm/40hlp/newfiles/ml3_3.year2.40hlp | premat > tempml33.dat

!cat /usr/users2/cm/40hlp/newfiles/i4_1.year1.40hlp | premat > templi41.dat
!cat /usr/users2/cm/40hlp/newfiles/i4_2.year1.40hlp | premat > templi42.dat
!cat /usr/users2/cm/40hlp/newfiles/i4_3.year1.40hlp | premat > templi43.dat

!cat /usr/users2/cm/40hlp/newfiles/i4_1.year2.40hlp | premat > temp2i41.dat
!cat /usr/users2/cm/40hlp/newfiles/i4_2.year2.40hlp | premat > temp2i42.dat
!cat /usr/users2/cm/40hlp/newfiles/i4_3.year2.40hlp | premat > temp2i43.dat

!cat /usr/users2/cm/40hlp/newfiles/i5_1.year2.40hlp | premat > tempi51.dat
!cat /usr/users2/cm/40hlp/newfiles/i5_2.year2.40hlp | premat > tempi52.dat
!cat /usr/users2/cm/40hlp/newfiles/i5_3.year2.40hlp | premat > tempi53.dat


load templg31.dat                %top instrument u,v,t,p
load templg32.dat
load templg33.dat

load templh31.dat
load templh32.dat
load templh33.dat

load temp2h31.dat
load temp2h32.dat
load temp2h33.dat
```



```
load temp2h61.dat
load temp2h63.dat

load tempml31.dat
load tempml32.dat
load tempml33.dat

load templi41.dat
load templi42.dat
load templi43.dat

load temp2i41.dat
load temp2i42.dat
load temp2i43.dat

load tempi51.dat
load tempi52.dat
load tempi53.dat

      t1g31 = templg31(:,2);
      t1g32 = templg32(:,2);
      t1g33 = templg33(:,2);

      t1h31 = templh31(:,2);
      t1h32 = templh32(:,2);
      t1h33 = [templh33(:,2); zeros((length(t1h31) length(templh33)),1)];

      t2h31 = temp2h31(:,2);
      t2h32 = temp2h32(:,2);
      t2h33 = temp2h33(:,2);

      t2h61 = temp2h61(:,2);
      t2h63 = temp2h63(:,2);

      tml31 = tempml31(:,2);
      tml32 = tempml32(:,2);
      tml33 = tempml33(:,2);

      t1i41 = templi41(:,2);
      t1i42 = templi42(:,2);
      t1i43 = templi43(:,2);

      t2i41 = temp2i41(:,2);
      t2i42 = temp2i42(:,2);
      t2i43 = temp2i43(:,2);

      ti51 = tempi51(:,2);
      ti52 = tempi52(:,2);
      ti53 = tempi53(:,2);

% make sure that CMs on each mooring start and end at same time

      nlg3 = 1:length(t1g32);
      nml3 = 1:length(tml32);
      nli4 = 1:length(t1i42);

%
% load temp regression matrix   [tCM1(p1) tCM2(p1+delp(2)) tCM3(p1+delp(3))]
% note: zero is the flag for no data, so can use 2 or 3 CMs on mooring
%

t = [ti51,ti52,ti53,
       t1h31, t1h32, t1h33, ...
```

```
        t2h31, t2h32, t2h33; ...
        tml31(nml3),tml32(nml3),tml33(nml3), ...
        tli41(nli4),tli42(nli4),tli43(nli4); ...
        t2i41 t2i42 t2i43;
        t2h61 zeros(t2h61) t2h63];
%        tlg31(nlg3),tlg32(nlg3),tlg33(nlg3)];

nlen = 1:length(t);

%
%set up parameters for regression, 3levels
%
%  Will fit 2 3 levels (p) of temp data to polynomial:
%  t(p) = cn(1)*(z+pl-p)^nc + cn(2)*(z+pl-p)^(nc-1) +...+ cn(nc)*(z+pl-p) + 12,
%
%  THUS, t(p=pref=z+pl) = 12 deg C  i.e.  pref = psuedo Z12 !!!!
%
%  where note: delp = -(pl - p(input data)) = [0 3.08 6.16];
%

nc=9;
err=.002;
delp=[0 3.075 2.03*3.075]; %press offsets from top instr. Diff than north!!!!
global z               % NOTE: p3-pl = 2.02*(p2-pl)  from mooring design.
global delp

np = length(t);

if(isempty(z))
        z=6+zeros(1,np);
        c0=[zeros(1,nc-1),1];%initial guess at coefficients for nc order
end

[cn,it]=moor(c0,t,err,10,1)     % include 5th argument if want fancy regression
zp=min(z)-max(delp):.1:max(z);
tp=polyval([cn 0],zp)+12;

plot(t(:,1),z,'w.',t(:,2),z-delp(2),'g.',t(:,3),z-delp(3),'b.',tp,zp,'+')

zsyn = z;

!rm temp* dat
```

```
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%
%                          MCOR.M
%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
function [cn,std,j]=mcor(cn,t,err,maxit,junk)
%[c,j]=mcor(c0,t,err,maxit) determines polynomial coefs, cn
%       cn = starting values  try [0 0 0 0 0 1] for 5th degree (6 coefs)
%       t = array of temperatures with zeros for no data
%       err = change in std from one iteration to next at which to stop
%       maxit = maximum # of iterations
std0=1.e35;
[m,n]=size(t);
nz=find(t~=0);
y=t(nz).^2;
np=length(cn);
nder=[np-1:-1:1];
for j=1:maxit
        tt=clock;
        for i=1:m
                [ct,nit]=nlreg('errbzp',[z(i) cn],t(i,:)',.005,20);
                z(i)=ct(1);
        end
        zj=z(ones(n,1),:)'-delp(ones(m,1),:);
        zj=zj(nz);

        a=ones(length(nz),np);
        a(:,np)=zj;
        for i=1:np-1
                a(:,np-i)=a(:,np-i+1).*a(:,np);
        end
        cn=(a\y)';

        if(nargin>4)
                [zz(1),nzz(1)]=max(zj),[zz(2),nzz(2)]=min(zj));
                for i=1:2
                        if(i==2)
                                nc=[1:2];
                        else
                                nc=find(polyval(cn.*nder,zz)<0);
                        end
                        lnc=length(nc);
                        if(lnc==0)
                                break
                        end
                        A=a'*a;
                        lncv=1:lnc;
                        A(:,np+lncv)=(a(nzz(nc),:).*nder(ones(lnc,1),:))';
                        A(np+lncv,1:np)=A(:,np+lncv)';
                        A(np+lncv,np+lncv)=zeros(lnc,lnc);
                        B=[a'*y;zeros(lnc,1)];
                        cnew=(A\B)';
                        cn=cnew(1:np);
                        if(lnc==2), break, end;
                end
        end

        e=y-a*cn';
        std=sqrt(e'*e/length(nz));
        fprintf('Iteration no. = %g, time = %g  Residual error = %g \n'
                ,j,fix(etime(clock,tt)),std)
%       !sleep 2
        if(abs(std-std0)<err)
                return
```



```
        end
        std0=std;
end
```

```
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%
%                                  NLREG.M
%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
function [cn,j]=nlreg(funct,c0,x,err,maxit)
%[c,j]=nlreg(funct,c0,x,err,maxit)
lambda=.001;
cn=c0;
[f,dfda]=feval(funct,cn,x);
chi0=f'*f;
beta=-(f'*dfda);
alpha=dfda'*dfda;
n=length(alpha);
for j=1:maxit
        alp=alpha+(tril(alpha)+triu(alpha)-alpha)*lambda;
        dc=(alp\beta')';
        cn(1:n)=c0(1:n)+dc;
%       [cn(1:n);dc]
        [f,dfda]=feval(funct,cn,x);
        chi1=f'*f;
%       stderr=sqrt(chi1/length(f));
%       keyboard
        if((min(err-abs(dc)))>0)
                return
        end
        if(chi1<chi0)
                beta=-(f'*dfda);
                alpha=dfda'*dfda;
                chi0=chi1;
                lambda=lambda/10;
                c0=cn;
        else
                lambda=lambda*10;
                cn=c0;
        end
end
```

```
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%
%                              ERRBZP.M
%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
function [f,fp]=errbzp(parms,t)
%[f fp]=errbp(parms,t) polynomial version for regression
nz=find(t~=0);
n=length(parms);
cp=parms(2:n)';
zj=(parms(1)-delp(nz))';
az=ones(length(zj),n);
az(:,n-1)=zj;
for i=2:n-1
        az(:,n-i)=az(:,n-i+1).*zj;
end
f=az(:,1:n-1)*cp-(t(nz)-12);
fp=az(:,2:n)*(cp.*[n-1:-1:1]');
```

```
CRASH$DKB0:[KAREN.TEX.MMOTION]MMCORN.M,2          4 FEB 1993 08:45          Page 1

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%
%                              MMCORN.M
%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% mmcorn.m   (similar to Nelson's moordrv.m)
% This program uses the northern canonical profile to correct
% a mooring's  temp and velocity to a given nominal pressure.
%
% This program is taylored to the SYNOP data. pnom = [400, 700, 1000].
% Special processing for records that don't start or end together
% is commented in.
%                                         Meghan Cronin  8/91
%
%   INPUT:  t, u, v, and pl and delta p on entire mooring
%             cm data: templ-4.dat    yrday hr t pcm(kPa) u(m/s) v
%           adcp data: tempa.dat      yrday hr tadcp padcp(m)
%      adcp binl data: tempb.dat      yrday hr tadcp pbinl(m) ubl(cm/s) vbl
%         NOTE: 1) premat a) strips headers, b) separates yrday-hr
%                  into yrday hr, and c) changes nodata flag to be 0.
%               2) i2_binl.year2.40hlp is in same units as cm data (SI units)
%           cn from synopbznorth.m (saved in north7.mat)
%           pmeas = measured pressure at top temperature used (cml or adcp).
%                   (pcml-pbin+2lm; pcml = padcp+12m; padcp = pbinl+9m etc.)
%                    BE CAREFUL ABOUT UNITS!
%                    TRANSFORM INTO UNITS OF 1000 kPa AT COMMENTS
%           fuldelp = [pl(top T)-pmeas, p2-pmeas, p3-pmeas] in 1000 kPa units
%                      Note: pmeas - pl - top temperature pressure
%           inst = nominal pressure level
%           pref = z12 when needed (if mooring has only 1 working temperature)
%           pcmoffset = offset between observed pressure and true p (1000 kPa)
%             pl = pmeas(1000 kPa) + pcmoffset(1000 kPa)
%
%
%   SUBROUTINES: zest.m
%                tempcor.m
%                tprefer.m
%                velc.m
%
%
%   OUTPUT: cor = [date hour tcor ucor vcor terr uerr verr]
%             z = regressed pref-pl (note: pref = zz12)
%            tr = t measured or simulated (to fill gaps) data
%
%   save session in e.g. h3_yrl.mat   in /usr/users/meghan/cm
%
%   save h3_400_yrl.dat cor -ascii  % in /usr/users/meghan/motiond/cmjan92/data
%
chdir /usr/users/meghan/motiond
load north7
global cn,
global terlkup;
global Cp,
global delp,
pcmoffset - 0,   %see individual cases when different ,
i2yr2-0,         %these will all be switched to 1 in ctrl section
h1yr2-0,         % when appropriate
adcpvel - 0,   %to correct binl velocity first you have to correct Tadcp up 9m
%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%%%%%%%%%%%%%%%%% DO NOT EDIT ABOVE THIS POINT %%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%
```

```
CRASH$DKB0:[KAREN.TEX.MMOTION]MMCORN.M,2          4 FEB 1993 08:45          Page 2

inst - 3          %  level to correct to
fuldelp - [0 (3.06+.12) ((2.04*3.06)+.12)];  % delta pressures between t1 t2 t3

%!cat /usr/users2/cm/40hlp/newfiles/h4 1.year1 40hlp | premat2 > templ.dat
!cat /usr/users2/cm/40hlp/newfiles/i2_2.year2.40hlp | premat2 > temp2.dat
!cat /usr/users2/cm/40hlp/newfiles/i2 3.year2 40hlp | premat2 > temp3 dat
!cat /usr/users2/cm/40hlp/newfiles/i2 4.year2.40hlp | premat2 > temp4.dat


%!cat /usr/users/cm/new.level.4/h2 4.year2.40hlp | premat2 > temp4.dat
pref-[]; tempb-[]; tempa-[];
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
templ - []; temp2 - []; temp3 - []; temp4 - [].

%load templ.dat
load temp2.dat
load temp3.dat
load temp4.dat
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%                                H3_YR1
%clear temp4   % Don't trust level 4 velocities  8/92
%!cat /usr/users2/cm/new.binl/h3yl b01.unc | prematb > tempb dat
%load tempb.dat
%pbinl - lowpass(tempb(:,4),40);    % tempb is unfiltered & sampled 1/hour
%pbinl - (pbinl(15:6:length(tempb))')/100;
%pcmoffset - -6/100;
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%                                H3 YR2
%!cat /usr/users2/cm/new.binl/h3y2 b01.unc | prematb > tempb dat
%load tempb dat
%pbinl - lowpass(tempb(:,4),40)'/100,          %put into 1000 kPa
%tadcp - lowpass(tempb(:,1),40);
%ubl - lowpass(tempb(:,2),40)/100;             %put into cm/s
%vbl - lowpass(tempb(:,3),40)/100;
%nt - 25:6:length(tempb);
%nt - nt(1:1748);             % tempb is longer than the CM data
%pbinl - pbinl(nt);  tadcp - tadcp(nt');  ubl-ubl(nt');  vbl - vbl(nt'),
%h3yr2-1;
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%                                H4_YR1
%load mid9
%!cat /usr/users2/cm/new.binl/h4yl b01.unc | prematb > tempb dat
% load tempb.dat
% pbinl - lowpass(tempb(:,4),40);
% pbinl - pbinl(15:6:length(tempb))'/100,
% pbinl - pbinl + .1;                    % pl was too small  Fudge factor is 10db
% load z12 lh4 oct92.dat                 % this is the latest and greatest
% pref - (z12 lh4 oct92(:,3)*(1010/1000)/100)',     %1010 db - 1000 m
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%                                H4 YR2   no tempa dat yet  1/92
% pcmoffset - 6/100,
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%                                I2 YR2
!cat /usr/users2/cm/new binl/i2 binl year2 40hlp | premat2 > tempb dat
 load tempb.dat
 i2yr2-1;                       % note  p - padcp
 adcpvel - 1;
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%                                I2 YR1 LEV 3 STARTS 1 PT LATE
% temp3 - [zeros(1,min(size(temp3))), temp3].
% pcmoffset - -6/100,
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%                                I1_YR2       DELETE LAST POINT OF ALL RECORDS
```

```
%n1 = 1:(length(temp1)-1);              %SINCE LOST LAST POINT OF P1
%temp1 = temp1(n1,:);
%temp2 = temp2(n1,:);
%temp3 = temp3(n1,:);
%temp4 = temp4(n1,:);
%clear n1
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%  FOR          13_YR1      MAKE TEMP1 = LENGTH(TEMP3)  I.E. DISCARD LAST DAY
%  temp1 = temp1(1:length(temp3),:);
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%%%%%%%%%%% THIS IS IT. DO NOT EDIT PAST THIS POINT. %%%%%%%%%%%%%%%%%%%%%%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

if(isempty(temp1))
   temp1 = tempb;
end

if(~isempty(temp2))
 if(temp2(1,2)~=temp1(1,2))
   error('do not start at same time')
end
elseif(~isempty(temp3))
  if(temp3(1,2)~=temp1(1,2))
   error('do not start at same time')
  end
end
if(inst == 1)
  z = [];
end

[n11, arg1] = size(temp1);
if (isempty(temp2))                    % This paragraph makes sure there are
   temp2 = zeros(n11,arg1);           % 3 datafiles. Otherwise makes one
end                                    % with all zeros.
if (isempty(temp3))
   arg3 = arg1-1;
   temp3 = zeros(n11,arg3);
end
if (isempty(temp4))
   arg3 = arg1 - 1;
   temp4 = zeros(n11,arg3);
end

[n14 arg4]= size(temp4);
[n12 arg2]= size(temp2);                         % This paragraph fills
[n13 arg3]= size(temp3);
nlen = max([n11,n12,n13]);                       % datafiles with zeros
nz1 = nlen-n11;                                  % so that all have same
nz2 = nlen-n12;                                  % length.
nz3 = nlen-n13;
nz4 = nlen-n14;
temp1 = [temp1; zeros(nz1,arg1)];
temp2 = [temp2; zeros(nz2,arg2)];
temp3 = [temp3; zeros(nz3,arg3)];
temp4 = [temp4; zeros(nz4,arg4)];

tyr = temp1(:,1);
thr = temp1(:,2);
thr2 = temp2(:,2);
thr3 = temp3(:,2);
thr4 = temp4(:,2);
```



```
tdr = [temp1(:,3) temp2(:,3), temp3(:,3)];

p1 = temp1(:,4)'./1000 + pcmoffset;    %pcmoffset is in units of 1000 kPa

if (~isempty(find(p1==0)))
   error('p1 has some zeros')
end

tstart2 = [temp2(1,1:2)];
tstart3 = [temp3(1,1:2)];    %  check that they really do all start together
tstart4 = [temp4(1,1:2)];

if(~isempty(pref))
  pref = [pref zeros(1:(length(temp1) length(pref)))];
end

if (arg3==5 & arg2==6);
  u = [temp1(:,5), temp2(:,5), temp3(:,4)];
  v = [temp1(:,6), temp2(:,6), temp3(:,5)];
elseif (arg3==6 & arg2==6);
  u = [temp1(:,5), temp2(:,5),temp3(:,5)];
  v = [temp1(:,6), temp2(:,6),temp3(:,6)];
elseif (arg3 ==5 & arg2==5)
  u = [temp1(:,5), temp2(:,4), temp3(:,4)];
  v = [temp1(:,6), temp2(:,5), temp3(:,5)];
else
  error('not filling up u and v right')
end

if(~isempty(tempb))
% if(tempb(1,1)~=temp2(1,1))
%    error('bin 1 does not start same time that temp2 does')
% end
  if(i2yr2 == 1 )                    % I2_yr2_bin1 data has SI units
     padcp = tempb(:,4)'/1000;
     pbin1 = (tempb(:,4)' - 9*10)/1000;
     p1 = padcp;
     tadcp = tempb(:,3);
     ub1 = tempb(:,5);
     vb1 = tempb(:,6);
  else
     if(length(pbin1)<length(tdr));
       nb = 1:length(pbin1);
     else
       nb = 1:length(tdr);
     end
     p1(nb) = pbin1(nb) + 21/100;
     padcp(nb) = pbin1(nb) + 9/100;
  end  %end if tempb is not emtpy & not i2 yr2
  if(length(pbin1)>length(temp2))
     nt = 1:length(u);
     pbin1 = pbin1(nt);
     if(adcpvel==1)
       tadcp = tadcp(nt');
       ub1 = ub1(nt');
       vb1 = vb1(nt');
     end  %end lengths of t u v bin1
  end  % end not same length for bin1
end  %end tempb not empty

%if(~isempty(tempa))  % 2h4              NOTE: THIS ADCP VELOCITY DIDN'T WORK
% if(tempa(1,1)~=temp2(1,1))            %      AND THE T,P WEREN'T PROCESSED
%    error('bin 1 does not start same time that temp2 does') %IN TIME TO
% end                                                        %INCORPORATE
```

```
% if(length(tempa) ~=length(temp2))                    %INTO MMOTION.
%    error('adcp is not same length as temp2')         %THEREFORE
% end                                                  %THE JAN92 H4 YR2
% padcp = tempa(:,4)'/100;                             %USED P1= PCM1-6M.
% pl = tempa(:,4)'./100 + 12/100;
%end

pnom = [4 7 10]'; %  corrected pressure levels
pnom = pnom(inst);

% CASES FOR MISSING TEMPERATURE DATA.
 ia = []; ib = []; ic = []; id = []; ie = [];            % CASES:    HAVE
 ia = find(tdr(:,1)~=0 & tdr(:,2)~=0 & tdr(:,3)~=0);    % case a: 1 2 3
 ib = find(tdr(:,1)~=0 & tdr(:,2)==0 & tdr(:,3)~=0);    % case b: 1   3
 ic = find(tdr(:,1)~=0 & tdr(:,2)~=0 & tdr(:,3)==0);    % case c: 1 2
 id = find(tdr(:,1)==0 & tdr(:,2)~=0 & tdr(:,3)~=0);    % case d:   2 3
 ie = find(tdr(:,1)~=0 & tdr(:,2)==0 & tdr(:,3)==0);    % case e: 1
 ig = find(tdr(:,1)==0 & tdr(:,2)==0 & tdr(:,3)~=0);    % case g:     3
     ica = [1 2 3];      ira = 2;
     icb = [1   3];      irb = 3;
     icc = [1 2  ];      irc = 2;
     icd = [  2 3];      ird = 2;
     ice = [1    ];      ire = 1;
     icg = [     3];     irg = 3;
nia = length(ia); nib = length(ib); nic = length(ic);
nid = length(id); nie = length(ie); nig = length(ig);

% COMPUTE REFERENCE PRESSURE FOR EACH GIVEN CASE; WILL DO REGRESSION
% AS WELL AS SOLVING THE ROOTS IF:   LENGTH(IC) > 1

if(isempty(z))               %if already have z then skip to feval polynomial

z=zeros(1,length(tdr));

        delp = fuldelp(ica);
        for i = 1:nia
                ii = ia(i);
                z(ii)= zest(tdr(ii,:),ira, ica,fuldelp);
        end

        delp = fuldelp(icb);
        for i = 1:nib
            ii = ib(i);
            if( tdr(ii,1)< 5.8 )            %this is for b6
                irb = 1; icb = 1;
            else
                irb = 3; icb = [1 3];
            end
            z(ii) = zest(tdr(ii,:),irb,icb,fuldelp);
        end

        delp = fuldelp(icc);
        for i = 1:nic;
            ii = ic(i);
                z(ii) = zest(tdr(ii,:),irc,icc,fuldelp);
        end

        delp = fuldelp(icd);
        for i = 1:nid
                ii = id(i);
                z(ii) = zest(tdr(ii,:),ird,icd,fuldelp);
        end

        delp = fuldelp(ice);
```



```
        for i = 1:nie
                ii = ie(i);
                z(ii) = zest(tdr(ii,:),ire,ice,fuldelp);
        end

        delp = fuldelp(icg);
        for i = 1:nig
                ii = ig(i);
                z(ii) = zest(tdr(ii,:),irg,icg,fuldelp);
        end

end                 % end calculating z

tpnom = zeros(nlen,1);
tc = zeros(nlen,1);
ter = zeros(nlen,1);

if(~isempty(pref))
  npre = find(pref~=0);
  zold = z;
  z(npre) = pref(npre)-pl(npre);
  plot(npre,zold(npre),npre,z(npre)),title('red-zold green-(z12/100)+pl')
end


if(max(z + pl - pnom ) >=max(zp) | min(z+pl pnom) <= min(zp) )
     disp('z out of range for pnom')
end

if(min(z fuldelp(1)) <= min(zp)  | max(z+fuldelp(3)))>= max(zp))
   disp('z out of range for t1 or t3')
end

tpn=polyval([cn 0], z+pl pnom )'+12;  %corrects temp to pnom

t1 = polyval([cn 0],z-fuldelp(1))' + 12;
t2 = polyval([cn 0],z fuldelp(2))' + 12;
t3 = polyval([cn 0],z-fuldelp(3))' + 12;
tsim = [t1 t2 t3];
nz = find(tdr(:,1:3)==0);
tr = tdr;
tr(nz)=tsim(nz);     %fill gappy CMs with simulated data
if(adcpvel==1| h)yr2==1);
  if(max(z+pl-pbin1) )= max(zp) )
       disp('z out of range for pbin')
       maxzpbin1 = max(z+pl pbin1)
       maxzp = max(zp)
  end
  tbin1sim = polyval([cn 0],z+pl-pbin1)' + 12;
  tadcpsim = polyval([cn 0],z+pl padcp)' + 12;
  tbin1 = tadcp + tbin1sim   tadcpsim;
  if(adcpvel==1)
     tr = [tbin1 tr(:,2:3)];
  end
end

%   COMPUTE ERROR IN TEMP FOR EACH GIVEN CASE  AND
%   FILL IN THE TEMP ARRAY WITH SIMULATED DATA FOR INTERPOLATING VEL

if(~isempty(ia))          % HAVE: 1 2 3
  iv = [1 2 3];
  delp = fuldelp(iv);
  [tc(ia),ter(ia)]=tempcor(tdr(ia,iv),tsim(ia,iv),tpn(ia),pl(ia),pnom z(ia));
end
```

```
if (~isempty(ib))          %  HAVE: 1   3
  iv = [1 3];
  delp = fuldelp(iv);
  [tc(ib),ter(ib)]=tempcor(tdr(ib,iv),tsim(ib,iv),tpn(ib),pl(ib),pnom,z(ib));
end
if (~isempty(ic))          %  HAVE: 1 2
  iv = [1 2];
  delp = fuldelp(iv);
  [tc(ic),ter(ic)]=tempcor(tdr(ic,iv),tsim(ic,iv),tpn(ic),pl(ic),pnom,z(ic));
end
if (~isempty(id))          %  HAVE:   2 3
  iv = [2 3];
  delp = fuldelp([2 3]);
  [tc(id),ter(id)]=tempcor(tdr(id,iv),tsim(id,iv),tpn(id),pl(id),pnom,z(id));
end
if (~isempty(ie))          %  HAVE: 1   MUST COMPUTE TC AND TER BY HAND
  if(inst~=1)
     tc(ie) = tpn(ie);
     tper = tprefer(tdr(ie,:),z(ie),pl(ie),pnom,fuldelp);
     tsc = tablel(terlkup,z(ie)+pl(ie)-pnom);
     ter(ie) = sqrt(tper.^2 + tsc.^2);
  else
     tc(ie) = tdr(ie,1) + tpn(ie) - tsim(ie,1);
     tperl = tprefer(tdr(ie,:),z(ie),pl(ie),pl(ie),fuldelp);
     tperpn = tprefer(tdr(ie,:),z(ie),pl(ie),pnom,fuldelp);
     scl = tablel(terlkup,z(ie));
     scp = tablel(terlkup,z(ie)+pl(ie)-pnom);
     ter(ie)= sqrt((.03)^2 + (scp-scl).^2 + (tperpn-tperl).^2);
  end

end
if (~isempty(ig))          %  HAVE:      3  MUST COMPUTE TC AND TER BY HAND
   if(inst~=3)
     tc(ig) = tpn(ig);
     tper = tprefer(tdr(ig,:),z(ig),pl(ig),pnom,fuldelp);
     tsc = tablel(terlkup,z(ig)+pl(ig)-pnom);
     ter(ig) = sqrt(tper.^2 + tsc.^2);
   else
     tc(ig) = tdr(ig,3) + tpn(ig) - tsim(ig,3);
     tperl = tprefer(tdr(ig,:),z(ig),pl(ig),pl(ig),fuldelp);
     tperpn = tprefer(tdr(ig,:),z(ig),pl(ig),pnom,fuldelp);
     scl = tablel(terlkup,z(ig)-fuldelp(3));
     scp = tablel(terlkup,z(ig)+pl(ig)-pnom);
     ter(ig)= sqrt((.03)^2 + (scp-scl).^2 + (tperpn-tperl).^2);
  end

end


tr = [tr temp4(:,3)];
u =  [u temp4(:,4)];
v =  [v temp4(:,5)];

plot(tr(:,1),z,'o',tr(:,2),z-fuldelp(2),'o',tr(:,3),z-fuldelp(3),'o',tp,zp)
title('click to move to next plot'), pause
plot(tc,z+pl-pnom,'x',tp,zp);
title('how well does tc-z fit on canonical profile?');

% FIND INDICES OF ALL CASES OF MISSING VEL DATA                      HAVE
  na = find(v(:,1)~=0 & v(:,2)~=0 & v(:,3)~=0);            % case a: 1 2 3
  nb = find(v(:,1)~=0 & v(:,2)==0 & v(:,3)~=0);            % case b: 1   3
  nc = find(v(:,1)~=0 & v(:,2)~=0 & v(:,3)==0);            % case c: 1 2
  nd = find(v(:,1)==0 & v(:,2)~=0 & v(:,3)~=0);            % case d:   2 3
  ne = find(v(:,1)~=0 & v(:,2)==0 & v(:,3)==0 & v(:,4)~=0);% case e: 1     4
```



```
  nf = find(v(:,1)==0 & v(:,2)~=0 & v(:,3)==0 & v(:,4)~=0);% case f:   2   4
  ng = find(v(:,1)==0 & v(:,2)==0 & v(:,3)~=0 & v(:,4)~=0);% case g:     3 4
%
if(~isempty(ng))
 error('PLEASE SEE MEGHAN. SHE DID NOT ACCOUNT FOR CASE G: ONLY 3 4 U&V')
end
%
% FIGURE OUT WHICH CMS TO USE IN INTERPOLATION FOR GIVEN CASE
% THEN COMPUTE VELOCITY AND ERROR FOR EACH CASE
% NOTE: FOR NORTHERN MOORINGS THAT DONT' MOVE MUCH, GOOD TO USE DEEP VEL
% TO INTERPOLATE TO 1000M
%
    Uc = -100*ones(nlen,2);       % elseif ONLY HAVE.  2 4; 3 4; -->> no vc
    erU = 1*ones(nlen,2);

%du = .02;   % This is the error in the CM velocity measurement
%dth =  5*pi/180;   % error in the angle to get shear


if(~isempty(na)),          % CASE A:  1 2 3
   ia = [1 2 3];      %case a: 1 2 3
   [Uc(na,:),erU(na,:)]=velc(tc(na),tr(na,ia),u(na,ia),v(na,ia),ter(na)),
end


if(~isempty(nb)),        %  CASE B: 1 3
   ib = [1 3],      %case b: 1 3
   [Uc(nb,:),erU(nb,:)]=velc(tc(nb),tr(nb,ib),u(nb,ib),v(nb,ib),ter(nb)),
end

if(~isempty(nc))        %  CASE C:  1 2
   if( ~isempty(find(tr(nc,4)==0)))
       error('bottom temp is zero for case c')
   end
   ic = [1 2];
   [Uc(nc,:),erU(nc,:)]=velc(tc(nc),tr(nc,ic),u(nc,ic),v(nc,ic),ter(nc)),
end

if(~isempty(nd))       %CASE D: 2 3
   if(~isempty(vbl))
        id = [1 2 3];    %case d(1): (binl) 2 3
        tr(nd,:)= [tbinl(nd) tr(nd,2:4)],
        u(nd,:) = [ubl(nd) u(nd,2:4)];
        v(nd,:) = [vbl(nd) v(nd,2:4)];
   else
        id = [2 3];
   end
   [Uc(nd,:),erU(nd,:)] = velc(tc(nd),tr(nd,id),u(nd,id),v(nd,id),ter(nd)),
end

if(~isempty(ne))          %CASE E: 1 4
    if(~isempty(find(tr(ne,4)==0)))
         error('some of the bottom temps are zero for case e')
    end
    ie = [1 4];        %case e: 1 4
    [Uc(ne,:),erU(ne,:)]=velc(tc(ne),tr(ne,ie),u(ne,ie),v(ne,ie),ter(ne)),
end

if (~isempty(nf))         %CASE F: 2 4
    if(~isempty(find(tr(nf,4)==0)))
       error('some of the bottom temps are zero for case f')
    end
    iff = [2 4];       %case f: 2 4
    [Uc(nf,:),erU(nf,:)]=velc(tc(nf),tr(nf,iff),u(nf,iff),v(nf,iff),ter(nf)),
```

```
end


%
%NOW MAKE ARRAY OF DATA CORRECTED FOR MOORING MOTION AND INCLUDE
%RANDOM MEASUREMENT ERROR FOR T (+ .03 deg) (U (+.02 m/s) already done)
nz = find(Uc==-100);
Uc(nz) = -1*(10^20)*ones(length(nz),1);
cor = [tyr thr tc Uc ter erU];
nlc = 1:length(cor);

% BLANK OUT WHEN NO T1 & T2 ISN'T IN THERMOCLINE. I DON'T TRUST THE ZREF.
%if(~isempty(id))   % don't have t1
%   nz=[];
%       nz = find( tdr(id,2)<5 );
%       if(~isempty(nz) & isempty(pref))
%               cor(id(nz),3:5) = -10^20*ones(length(nz),3);
%               cor(id(nz),6:8) = ones(length(nz),3);
%       end
%end
%if(~isempty(ig))   % ONLY T3
%nz = [];
%       nz = find( tdr(ig,3)< 4.8 );
%       if(~isempty(nz)& isempty(pref))
%               cor(ig(nz),3:5) = -10^20*ones(length(nz),3);
%               cor(ig(nz),6:8) = ones(length(nz),3);
%       end
%end
% BLANK OUT WHEN HAVE TO EXTRAPOLATE MORE THAN 100M AND WHEN NO U1  &
% U2 IS NOT IN THE THERMOCLINE. NOTE: 1000M IS NOT EXTRAPOLATION IN THIS CASE.


if(~isempty(nd))
 nz = [];
        nz = find(p1(nd)'>7.35); % don't use p2>1040 & p3>1350 to extrap to 4
        if(~isempty(nz) & inst==1)
        disp('dont use p2>1040 & p3>1350 to extrap to 4: applies to these pts')
                size(nz)
                cor(nd(nz),4:5) = -10^20*ones(length(nz),2);
                cor(nd(nz),7:8) = ones(length(nz),2);
        end
end

if(~isempty(nf))    % don't have u1 or u3:
nz = [];
      nz = find(tr(nf,2)<5.6 & p1(nf)'>5); %T2 IS BELOW THERMOCLINE & MUST EXTR
      if(~isempty(nz) & inst~=3)
          disp('t2<5.6 below thermo& p1>5 big extrap & do not have u1 or u3')
          disp('therefore do not correct u,v 400 or 700 for this many points:')
                size(nz)
                cor(nf(nz),4:5) = -10^20*ones(length(nz),2);
                cor(nf(nz),7:8) = ones(length(nz),2);
      end
end

nz = find(cor(:,5)~=-10^20);

clear nit ii ct  zmax zmin arg1 arg2 arg3 arg4 vr ur tt sc zr thr2
clear nia nib nic nid nie i uer ver sn1 sn3 nl1 nl2 nl3 nl4 nlen Uc tpn
clear nz1 nz2 nz3 nz4 tsim tyr thr iv ia ib ic icg id ie irg na tpnom
clear na4 nb nb4 nc nd ne nf ng nt nzt tstart1 tstart2 tstart3 tstart4
clear ica icb icc icd ice ira irb irc ird ire lev dth  dum dum1 erU
clear ig nig sim1 sim2 sim3 terr du t1 t2 t3 tc ter nb adcpvel delpsyn
clear thr1 thr4 temp1 temp2 temp3 temp4 tempa tempb tper1 tperpn npre
```



```
clear i2yr2 npts scl scp t tsc tper zsyn it tbinlsim tadcpsim
!rm temp*.dat
```

```
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%
%                              MMCORS.M
%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%  mmcors1234 jan93.m  %                    by K. Tracey
%   Updated version of Meghan's mmcors.m. %   Revisions were needed to use
% the data files stored in the database. Note the awk program db2mmcor is
% used to create dummy column for time.
%
%  The basic code of this m file is Meghan's code for processing done
% prior to Nov 1992. Thus it does uses all 4 levels of current meter data.
%
%    Also sets a flag for the M13 mooring so that the level 2 measured
% pressures are used as  p1 rather than the level 1 pressures.  Thus
% fuldelp is adjusted to be the differences between level 2 and levels 1 and
% 3.
% ######################################################################

% mmcors.m   (similar to Nelson's mcoodrv.m)
% This program uses the southern canonical profile to correct
% a mooring's  temp and velocity to a given nominal pressure.
%
% This program is taylored to the SYNOP data. pnom = [400, 700, 1000].
% Special processing for records that don't start or end together
% is commented in.
%                                   Meghan Cronin  8/91
%
%
%   INPUT:  t, u, v, and p1 on entire mooring (temp1.dat - temp4.dat)
%                 NOTE: premat is an awk program which strips
%                       headers and  changes nodata flag = 0.
%           cn from synopbzsouth.m (saved in mid9.mat)
%           fuldelp = [0, p2-p1, p3-p1] NOTE: when using ADCP
%                   data, bin1 p (& v) is 12 m above the temperature sensor
%           p1 =  pressure of top temperature.
%           adcpvel = 0 if not using ADCP velocity,
%                   = 1 if using ADCP velocity. Because the ADCP vel is 12 m
%                      above the T, make sure to correct T1 up to ADCP
%                      level before correcting the velocity
%           inst = nominal pressure level
%
%   SUBROUTINES: zest.m
%                tempcor.m
%                tprefer.m
%                velc.m
%
%
%   OUTPUT: cor = [date hour toor uoor voor terr uerr verr]
%             z = regressed pref-p1
%            tr = t measured or simulated (to fill gaps) data
%
%   save session in e.g. g1_yr1.mat   in /usr/users/meghan/motiond/cmjan92
%
%   save g1 1 yr1.dat cor  ascii  % in /usr/users/meghan/motiond/cmjan92/data
load mid9
global cn
global Cp,
global delp,
global teclkup,

%### Note for M13 use level  2  instead of level 1 as upper level.
%### So this flag, m13yr2,  gets changed in the m13 control section below.
m13yr2=0,
```



```
inst = 3        %  level to correct to

% ######################################################################
% Below is Meghan's original code which is commented out
%fuldelp   [0 1.09 2.02*3.09];    %  delta pressures between instruments


%!cat /usr/users2/cm/40hlp/newfiles/h6 1.year2.40hlp | premat2 > temp1.dat
%!cat /usr/users2/cm/40hlp/newfiles/h6 2.year2.40hlp | premat2 > temp2.dat
%!cat /usr/users2/cm/40hlp/newfiles/h6 3.year2.40hlp | premat2 > temp3.dat
%!cat /usr/users2/cm/40hlp/newfiles/h6 4.year2.40hlp | premat2 > temp4.dat
%!cat /usr/users2/cm/new.level.4/q2 4.year2.40hlp | premat2 > temp4.dat

% The following is the update for data base
fuldelp = [-3.08 0 1.01*3.08];    %  delta pressures between instruments

!cat /usr/users/db/db/cm/89 90c/lev1/m13 | db2mmcor > temp1.dat
!cat /usr/users/db/db/cm/89 90c/lev2/m13 | db2mmcor > temp2.dat
!cat /usr/users/db/db/cm/89 90c/lev3/m13 | db2mmcor > temp3.dat
!cat /usr/users/db/db/cm/89 90c/lev4/m13 | db2mmcor > temp4.dat
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%  !!!!!!!   CHECK:  G2 YR2  G3 YR1,  I3 YR2,   I4 YR1  M11 YR2  !!!!!!!!!!!!!
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

temp1 = []; temp2 = [], temp3 = []; temp4   [], pref = [], tempb   [],
tempa = [];
load temp1.dat
load temp2.dat
load temp3.dat
load temp4.dat

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%                                G2_YR2
% load z12_2g2.dat
% pref = (z12_2g2(:,3)*(1.01)/100)';
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% FOR G3_YR1: DELETE 1ST POINT OF 2 & 4  SO ALL START TOGETHER
%  temp2 = temp2(2:length(temp2),:),
%  temp4 = temp4(2:length(temp4),:),
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% FOR I4_YR1: DELETE END OF LEVEL 3 SINCE 1 & 2 ARE SHORTER
%  temp3 = temp3(1:length(temp1),:);
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% FOR I3_YEAR2: TEMP ON LEVEL 2 GOT BAD TOWARDS END    FILL WITH ZEROS
%  temp2 =[temp2(:,1:2) [temp2(1:1230,3);zeros(165,1)] temp2(:,4:6)];
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%  FOR H6 YR2: USE IES PREF UNTIL GET BETTER T3 DATA
% load z12 h6.dat
% pref = z12 h6(:,3)'*(1.01)/100,
% temp1 = temp1(72:71+length(pref),:),
% temp3 = temp3(72:71+length(pref),:),
% temp4 = temp4(72:71+length(pref),:),
% temp3(:,3) = zeros(length(temp1),1),

% ######################################################################
% FOR M13: USE FULDELP RELATIVE TO LEVEL2 INSTEAD OF LEVEL1
%           SET FLAG FOR LATER USE WHEN DEFINING P1
fuldelp = [-3.08 0 1.01*3.08],    %  delta pressures between instruments
m13yr2=1, %flag set for defining p1 later on
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
```

```
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%%%%%%%%%%%%%%%%%% DO NOT EDIT PAST THIS POINT!!!! %%%%%%%%%%%%%%%%%%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%
if(~isempty(temp2))
 if(temp2(1,2)~=temp1(1,2))
   error('do not start at same time')
end
elseif(~isempty(temp3))
  if(temp3(1,2)~=temp1(1,2))
   error('do not start at same time')
  end
end
if(inst == 1)
  z = [];
end

[nl1, arg1] = size(temp1);
if (isempty(temp2))                  % This paragraph makes sure there are
   temp2 = zeros(nl1,arg1);         % 3 datafiles. Otherwise makes one
end                                 % with all zeros.
if (isempty(temp3))
   arg3 = arg1-1;
   temp3 = zeros(nl1,arg3);
end
if (isempty(temp4))
   arg3 = arg1 - 1;
   temp4 = zeros(nl1,arg3);
end

[nl4 arg4]= size(temp4);
[nl2 arg2]= size(temp2);                          % This paragraph fills
[nl3 arg3]= size(temp3);
nlen = max([nl1,nl2,nl3]);                        % datafiles with zeros
nz1 = nlen-nl1;                                   % so that all have same
nz2 = nlen-nl2;                                   % length.
nz3 = nlen-nl3;
nz4 = nlen-nl4;
temp1 = [temp1; zeros(nz1,arg1)];
temp2 = [temp2; zeros(nz2,arg2)];
temp3 = [temp3; zeros(nz3,arg3)];
temp4 = [temp4; zeros(nz4,arg4)];

tyr = temp1(:,1);
thr = temp1(:,2);
thr2 = temp2(:,2);
thr3 = temp3(:,2);
thr4 = temp4(:,2);

tdr = [temp1(:,3) temp2(:,3), temp3(:,3)];

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% Revised in Jan 93 by K. Tracey to include the ml3
% condition for using p2 as desired level

if (ml3yr2 == 1)
  p1 = temp2(:,4)'/1000;
else
  p1 = temp1(:,4)'/1000;
end
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

if (~isempty(find(p1==0)))
```



```
    error('p1 has some zeros')
end

tstart2 = [temp2(1,1:2)];
tstart3 = [temp3(1,1:2)];    %  check that they really do all start together
tstart4 = [temp4(1,1:2)];

if(~isempty(pref))
  pref = [pref zeros(1:(length(temp1) length(pref)))];
end

if (arg3==5 & arg2==6);
  u = [temp1(:,5), temp2(:,5), temp3(:,4)];
  v = [temp1(:,6), temp2(:,6), temp3(:,5)];
elseif (arg3==6 & arg2==6);
  u = [temp1(:,5), temp2(:,5),temp3(:,5)];
  v = [temp1(:,6), temp2(:,6),temp3(:,6)];
elseif (arg3 ==5 & arg2==5)
  u = [temp1(:,5), temp2(:,4), temp3(:,4)];
  v = [temp1(:,6), temp2(:,5), temp3(:,5)];
else
  error('not filling up u and v right')
end

pnom = [4 7 10]', %  corrected pressure levels
pnom = pnom(inst);

% CASES FOR MISSING TEMPERATURE DATA.
 ia = [], ib = []; ic = [], id = []; ie = [],          % CASES      HAVE
 ia = find(tdr(:,1)~=0 & tdr(:,2)~=0 & tdr(:,3)~=0),    % case a: 1 2 3
 ib = find(tdr(:,1)~=0 & tdr(:,2)==0 & tdr(:,3)~=0),    % case b: 1   3
 ic = find(tdr(:,1)~=0 & tdr(:,2)~=0 & tdr(:,3)==0),    % case c: 1 2
 id = find(tdr(:,1)==0 & tdr(:,2)~=0 & tdr(:,3)~=0),    % case d:   2 3
 ie = find(tdr(:,1)~=0 & tdr(:,2)==0 & tdr(:,3)==0),    % case e: 1
 ig = find(tdr(:,1)==0 & tdr(:,2)==0 & tdr(:,3)~=0),    % case g:     3
     ica = [1 2 3];     ira = 2;
     icb = [1   3];     irb = 3;
     icc = [1 2  ];     irc = 2;
     icd = [  2 3];     ird = 2;
     ice = [1    ];     ire = 1;
     icg = [     3];    irg = 3;
nia = length(ia); nib = length(ib); nic = length(ic);
nid = length(id); nie = length(ie); nig = length(ig);

% COMPUTE REFERENCE PRESSURE FOR EACH GIVEN CASE  WILL DO REGRESSION
% AS WELL AS SOLVING THE ROOTS IF:   LENGTH(IC) > 1

if(isempty(z))               %if already have z then skip to feval polynomial

z=zeros(1,length(tdr));

        delp = fuldelp(ica),
        for i = 1:nia
                ii = ia(i);
                z(ii)= zest(tdr(ii,:),ira, ica,fuldelp),
        end

        delp = fuldelp(icb),
        for i = 1:nib
            ii = ib(i),
            if( tdr(ii,1)< 5.0 )          %this is for h6
                irb = 1; icb = 1,
            else
                irb = 1, icb = [1 3],
```

```
        end
        z(ii) = zest(tdr(ii,:),irb,icb,fuldelp);
    end

    delp = fuldelp(icc);
    for i = 1:nic;
        ii = ic(i);
            z(ii) = zest(tdr(ii,:),irc,icc,fuldelp);
    end

    delp = fuldelp(icd);
    for i = 1:nid
            ii = id(i);
            z(ii) = zest(tdr(ii,:),ird,icd,fuldelp);
    end

    delp = fuldelp(ice);
    for i = 1:nie
            ii = ie(i);
            z(ii) = zest(tdr(ii,:),ire,ice,fuldelp);
    end

    delp = fuldelp(icg);
    for i = 1:nig
            ii = ig(i);
            z(ii) = zest(tdr(ii,:),irg,icg,fuldelp);
    end

end                 % end calculating z

tpnom = zeros(nlen,1);
tc = zeros(nlen,1);
ter = zeros(nlen,1);

if(~isempty(pref))
  npre = find(pref~=0);
  zold = z;
  z(npre) = pref(npre)-pl(npre);
  plot(npre,zold(npre),npre,z(npre));title('red=zold green=(z12/100)+pl')
end

if(min(z-fuldelp(3)) <= min(zp) )
   disp('z out of range for t3')
end
if( max(z-fuldelp(1)) >= max(zp))
   disp('z out of range for t1 ')
end
if(max(z + pl - pnom ) >= max(zp) | min(z+pl-pnom) <= min(zp) )
      disp('z out of range for pnom')
end

tpn=polyval([cn 0], z+pl-pnom )'+12;  %corrects temp to pnom

t1   polyval([cn 0],z-fuldelp(1))' + 12;
t2 = polyval([cn 0],z-fuldelp(2))' + 12;
t3   polyval([cn 0],z-fuldelp(3))' + 12;
tsim = [t1 t2 t3];
nz = find(tdr(:,1:3)==0);
tr = tdr;
tr(nz)=tsim(nz);     %fill gappy CMs with simulated data

%    COMPUTE ERROR IN TEMP FOR EACH GIVEN CASE  AND
%    FILL IN THE TEMP ARRAY WITH SIMULATED DATA FOR INTERPOLATING VEL
```



```
if(~isempty(ia))          % HAVE: 1 2 3
  iv = [1 2 3];
  delp = fuldelp(iv);
  [tc(ia),ter(ia)]=tempcor(tdr(ia,iv),tsim(ia,iv),tpn(ia),pl(ia),pnom,z(ia));
end
if (~isempty(ib))         %  HAVE: 1   3
  iv = [1 3];
  delp = fuldelp(iv);
  [tc(ib),ter(ib)]=tempcor(tdr(ib,iv),tsim(ib,iv),tpn(ib),pl(ib),pnom,z(ib));
end
if (~isempty(ic))        %   HAVE: 1 2
  iv = [1 2];
  delp = fuldelp(iv);
  [tc(ic),ter(ic)]=tempcor(tdr(ic,iv),tsim(ic,iv),tpn(ic),pl(ic),pnom,z(ic));
end
if (~isempty(id))        %   HAVE:   2 3
  iv = [2 3];
  delp = fuldelp([2 3]);
  [tc(id),ter(id)]=tempcor(tdr(id,iv),tsim(id,iv),tpn(id),pl(id),pnom,z(id));
end

if (~isempty(ie))        %   HAVE:  1   MUST COMPUTE TC AND TER BY HAND
ns = [];
     if(inst==1)
         ns = find(pl(ie)>4.75);
     else
         ns = 1:length(ie);
     end
     if(inst == 1)
       tc(ie) = tdr(ie,1) + tpn(ie)   tsim(ie,1);
       tperl = tprefer(tdr(ie,:),z(ie),pl(ie),pl(ie),fuldelp);
       tperpn = tprefer(tdr(ie,:),z(ie),pl(ie),pnom,fuldelp);
       scl = tablel(terlkup,z(ie));
       scp = tablel(terlkup,z(ie)+pl(ie) pnom);
       ter(ie)= sqrt((.03)^2 + (scp scl).^2 + (tperpn tperl) ^2);
     end
     if(~isempty(ns))
         tc(ie(ns)) = tpn(ie(ns));
         tper = tprefer(tdr(ie(ns),:),z(ie(ns)),pl(ie(ns)),pnom,fuldelp);
         tsc = tablel(terlkup,z(ie(ns))+pl(ie(ns)) pnom);
         ter(ie(ns)) = sqrt(tper.^2 + tsc.^2);
     end    %end just use the profile tc. if extrap more than 75 m or so
end   %end is case ie empty

if (~isempty(ig))        %   HAVE:      3  MUST COMPUTE TC AND TER BY HAND
   if(inst~=3)
     tc(ig) = tpn(ig);
     tper = tprefer(tdr(ig,:),z(ig),pl(ig),pnom,fuldelp);
     tsc = tablel(terlkup,z(ig)+pl(ig) pnom);
     ter(ig) = sqrt(tper.^2 + tsc.^2);
   else
     tc(ig) = tdr(ig,3) + tpn(ig)   tsim(ig,3);
     tperl = tprefer(tdr(ig,:),z(ig),pl(ig),pl(ig),fuldelp);
     tperpn = tprefer(tdr(ig,:),z(ig),pl(ig),pnom,fuldelp);
     scl = tablel(terlkup,z(ig) fuldelp(3));
     scp = tablel(terlkup,z(ig)+pl(ig) pnom);
     ter(ig)= sqrt((.03)^2 + (scp scl) ^2 + ((tperpn tperl) ^2);
  end

end


tr = [tr temp4(:,3)];
u = [u  temp4(:,4)];
```

```
v = [v  temp4(:,5)];

%plot(tr(:,1),z,'o',tr(:,2),z-fuldelp(2),'o',tr(:,3),z-fuldelp(3),'o',tp,zp)
plot(tc,z+pl-pnom,'x',tp,zp);
title('how well does tc-z fit on canonical profile?');

% FIND INDICES OF ALL CASES OF MISSING VEL DATA                        HAVE
  na = find(v(:,1)~=0 & v(:,2)~=0 & v(:,3)~=0 & v(:,4)==0);% case a: 1 2 3
  na4= find(v(:,1)~=0 & v(:,2)~=0 & v(:,3)~=0 & v(:,4)~=0);% case a4:1 2 3 4
  nb = find(v(:,1)~=0 & v(:,2)==0 & v(:,3)~=0 & v(:,4)==0);% case b: 1   3
  nb4= find(v(:,1)~=0 & v(:,2)==0 & v(:,3)~=0 & v(:,4)~=0);% case b4:1   3 4
  nc = find(v(:,1)~=0 & v(:,2)~=0 & v(:,3)==0 & v(:,4)==0);% case c: 1 2
  nc4= find(v(:,1)~=0 & v(:,2)~=0 & v(:,3)==0 & v(:,4)~=0);% case c4:1 2   4
  nd = find(v(:,1)==0 & v(:,2)~=0 & v(:,3)~=0 & v(:,4)==0);% case d:   2 3
  ne = find(v(:,1)~=0 & v(:,2)==0 & v(:,3)==0 & v(:,4)~=0);% case e: 1     4
  nf = find(v(:,1)==0 & v(:,2)~=0 & v(:,3)==0 & v(:,4)~=0);% case f:   2   4
  ng = find(v(:,1)==0 & v(:,2)==0 & v(:,3)~=0 & v(:,4)~=0);% case g:     3 4;

%
  if(~isempty(ng))
    error('PLEASE SEE MEGHAN. SHE DID NOT ACCOUNT FOR THIS CASE')
  end
%
% FIGURE OUT WHICH CMS TO USE IN INTERPOLATION FOR GIVEN CASE AND PNOM
% THEN COMPUTE SHEAR, VELOCITY AND ERROR FOR EACH CASE
%
    Uc =  100*ones(nlen,2);        % elseif ONLY HAVE:  2 4; 3 4; -->> no vc
    erU = 1*ones(nlen,2);
    uerr = 1*ones(nlen,1);
    verr = 1*ones(nlen,1);

%du = .02,   %  This is the error in the CM velocity measurement
%dth =  5*pi/180;   %  error in the angle to get shear  These are in velc.m


if(~isempty(na)),            % CASE A:  1 2 3
   ia = [1 2 3];       %case a: 1 2 3
   [Uc(na,:),erU(na,:)]=velc(tc(na),tr(na,ia),u(na,ia),v(na,ia),ter(na));
end

if(~isempty(na4)),            % CASE A4:  1 2 3 4
  if(~isempty(find(tr(na4,4)==0)))
      error('some of the bottom temps are zero for case a4')
  end
   ia = [1 2 3 4];       %case a4: 1 2 3 4
  [Uc(na4,:),erU(na4,:)]=velc(tc(na4),tr(na4,ia),u(na4,ia),v(na4,ia),ter(na4));
end


if(~isempty(nb)),         %  CASE B: 1 3
   ib = [1 3],       %case b: 1 3
   [Uc(nb,:),erU(nb,:)]=velc(tc(nb),tr(nb,ib),u(nb,ib),v(nb,ib),ter(nb));
end

if(~isempty(nb4)),          %  CASE B4: 1 3 4
  if(~isempty(find(tr(nb4,4)==0)))
      error('some of the bottom temps are zero for case b4')
  end
  ib = [1 3 4],       %case b4: 1 3 4
 [Uc(nb4,:),erU(nb4,:)]=velc(tc(nb4),tr(nb4,ib),u(nb4,ib),v(nb4,ib),ter(nb4));
end

if(~isempty(nc))         %  CASE C:  1 2
```



```
   ic = [1 2];
   [Uc(nc,:),erU(nc,:)]=velc(tc(nc),tr(nc,ic),u(nc,ic),v(nc,ic),ter(nc));
end

if(~isempty(nc4))       %  CASE C4:  1 2 4
  if(~isempty(find(tr(nc4,4)==0)))
      error('some of the bottom temps are zero for case c4')
  end
   ic = [1 2 4];
 [Uc(nc4,:),erU(nc4,:)]=velc(tc(nc4),tr(nc4,ic),u(nc4,ic),v(nc4,ic),ter(nc4)),
end

if(~isempty(nd))        %CASE D: 2 3
        id = [2 3];
  [Uc(nd,:),erU(nd,:)] = velc(tc(nd),tr(nd,id),u(nd,id),v(nd,id),ter(nd)),
end


if(~isempty(ne))           %CASE E: 1 4
    if(~isempty(find(tr(ne,4)==0)))
         error('some of the bottom temps are zero for case e')
    end
    ie = [1 4];          %case e: 1 4
    [Uc(ne,:),erU(ne,:)]=velc(tc(ne),tr(ne,ie),u(ne,ie),v(ne,ie),ter(ne)),
end

if (~isempty(nf))           %CASE F: 2 4
    if(~isempty(find(tr(nf,4)==0)))
         error('some of the bottom temps are zero for case f')
    end
    iff = [2 4],          %case f: 2 4
    [Uc(nf,:),erU(nf,:)]=velc(tc(nf),tr(nf,iff),u(nf,iff),v(nf,iff),ter(nf)),
end
%
%NOW MAKE ARRAY OF DATA CORRECTED FOR MOORING MOTION AND INCLUDE
%RANDOM MEASUREMENT ERROR FOR T (+ .03 deg) (U (+.02 m/s) already done)
nz = find(Uc==-100);
Uc(nz) = -1*(10^20)*ones(length(nz),1),
cor = [tyr thr tc Uc ter erU];
nlc = 1:length(cor);

% BLANK OUT WHEN NO T1 & T2 ISN'T IN THERMOCLINE. I DON'T TRUST THE ZREF

if(~isempty(id))   % don't have t1
   nz=[];
        nz = find( tdr(id,2)<5 );
        if(~isempty(nz))
         disp('t2 is below thermocline <5 &do not have t1'),
         disp('therefore, will blank out t,u,v for this many days ')
                length(nz)
                cor(id(nz),3:5) =  10^20*ones(length(nz),1),
                cor(id(nz),6:8) = ones(length(nz),1),
        end
end
if(~isempty(ig))   % ONLY T3
nz = [],
        nz = find( tdr(ig,1)< 4.8 & isempty(pref) ),
        if(~isempty(nz))
         disp('t3 is below thermocline <4.8 &do not have t1 or t2'),
         disp('therefore, will blank out t,u,v for this many days ')
                length(nz)
                cor(ig(nz),3:5) =  10^20*ones(length(nz),1),
                cor(ig(nz),6:8) = ones(length(nz),1),
        end
```

```
end

% BLANK OUT WHEN HAVE TO EXTRAPOLATE MORE THAN 100M AND WHEN NO U1  &
% U2 IS NOT IN THE THERMOCLINE. NOTE: 1000M IS NOT EXTRAPOLATION IN THIS CASE.


if(~isempty(nd))
 nz = [];
        nz = find(p1(nd)'>7.35); % don't use p2>1040 & p3>1350 to extrap to 4
        if(~isempty(nz) & inst==1)
         disp('p1 is below 735 db, & do not have u1');
         disp('therefore, will blank out u400&v400 for this many days:')
                length(nz)
                cor(nd(nz),4:5) = -10^20*ones(length(nz),2);
                cor(nd(nz),7:8) = ones(length(nz),2);
        end
end

if(~isempty(nf))    % don't have u1 or u3:
nz = [];
      nz = find(tr(nf,2)<5.6 & p1(nf)'>5); %T2 IS BELOW THERMOCLINE & MUST EXTR
      if(~isempty(nz) & inst~=3)
        disp('p1 is below 500, t2 is below thermo <5.6 & donot have u1 or u3');
        disp('therefore, will blank out u,v 400 & 700 for this many days:')
                length(nz)
                cor(nf(nz),4:5) = -10^20*ones(length(nz),2);
                cor(nf(nz),7:8) = ones(length(nz),2);
      end
end

nz = find(cor(:,5)~=-10^20);
n1 = find(cor(:,3)==-10^20);

clear nit ii ct  zmax zmin arg1 arg2 arg3 arg4 vr ur tt sc zr
clear nia nib nic nid nie i uer ver sn1 sn3 n11 n12 n13 n14 nlen Uc
clear nz1 nz2 nz3 nz4 temp1 temp2 temp3 temp4 tsim tyr thr iv uerr verr
clear ica icb icc icd ice ira irb irc ird ire lev dth  dum dum1
clear ig nig sim1 sim2 sim3 terr  y dth13 dth14 dth24 dth23 du iiv
clear na na4 nb nb4 nc nc4 nd nd4 ne nf ng iff delpsyn zsyn erU npts
clear thr2 thr3 thr4 tstart2 tstart3 t t1 t2 t3 tc tper1 tperpn
clear tpn tpnom nzt ia ib ic icg id ie irg scl scp
clear ter tempb tempa tstart4
!rm temp1.dat temp2.dat temp3.dat
```

```
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%
%                              TEMPCOR.M
%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%
%  This function is called by mmcors.m and mmcorn.m to correct the
%  temperature using the simulated and observed temperatures at
%  at the CM levels, and using the canonical profile's nominal
%  pressure temperature.
%
%  INPUTS:
%      T    - WORKING observed temperature  (NCM >=2)
%      Tsim - WORKING simulated temperature
%      Tpnom - canonical profile temperature at nominal pressure
%      pl    - level 1 pressure
%      z     - pref - pl
%
%  OUTPUTS:
%      tc - corrected temperature
%      ter - error in corrected temperature
%
%  IN DRIVER PROGRAM: define: delp = fuldelp(iv)  prior to call
%
%  GLOBAL VARIABLES: terlkup, cn, Cp, delp
% ----------------------------------------------------------------------
%
%  THEORY:  tc = w1*tc1 + w2*tc2   if interpolating or extrapolating < 100m
%              or   tc = tpro(pn)   if extrapolating more than 100m
%
%  where:   w1 = |p2-pnom|/(|pnom-pl| + |p2-pnom|)
%    and    w2 = |pnom-pl|/(|pnom-pl| + |p2-pnom|) = 1-w1;
%
%   and,    tc1 = T(1) + (Tpnom - Tsim(1))
%           tc2 = T(2) + (Tpnom - Tsim(2))
%
% This formulation has the property that when pnom = pl or p2, tc = T observed
% Basically it starts at the 2 nearest Tp measurements and goes up (or down)
% to pnom along a curve which is parallel to the canonical profile. It
% arrives at 2 (different) 'corrected' tempeeratures, tc1 and tc2, which
% it weights according to the distance between the CMs and the nominal
% pressure.
%
%        NOTE: THIS PROGRAM REQUIRES AT LEAST 2 WORKING CMS.
%      in case e or g ( e == CM1 only,  g == CM3 only), then
%      the tc and ter must be computed by hand.
%      IF the  nominal level is  !!NOT!!! the working CM,
%       then tc = canocial profile and ter = sqrt(sc^2 + (dtdz*prefer)^2)
%      IF the nominal level -- the only working level,
%         then  tc = TOBS + (TPNOM - TSIM)
%
%      THUS, CASE E OR G SHOULD COMPUTE TC AND ERROR WITHIN
%      MMCORN.M OR MMCORS.M  (DO NOT USE THIS PROGRAM).
%
%      THE MOORING MOTION CORRECTION REQUIRES THAT PREF PNOM LIE
%        ON THE CANONICAL PROFILE. IF IT DOESN'T, THIS PROGRAM
%        WILL BOMB
%
%      ALSO: THE STRETCHING REQUIRES THAT PREF PU,PL LIE ON THE
%       CANONICAL PROFILE. i.e. require z+pl-pu < max(terlkup(:,1)
%       and  z+pl-pl > min(terlkup(:,1) . If these conditions are
%       not satisfied, then TPNOM is used.
%
%. The error in the corrected temperature, ter, is due to the error in the
```



```
%  thermistors, dt = .03 deg C, and pressure, dp= 05, the scatter around the
%  canonical profile (from terlkup), and the error in tc due an error in the
%  pref.  It is assumed that the scatter at the upper CM is of the
%  same sign as the scatter at the nominal pressure, and likewise the
%  scatter at the lower CM is of same sign as the scatter at the nominal
%  pressure.
%
%
%  Altogether, using weights:
%  ---------------------------------------------------------------------
% |err(tc) = sqrt( (w1^2 + w2^2)dt^2
%                + [(tc1 tc2)(w1*dp/(|pl pnom| + |pu pnom|))]^2
%                + [(tc1 tc2)(w2*dp/(|pl pnom| + |pu pnom|))]^2
%                +  w1^2[sc(pnom) - sc(1)]^2   + w2^2[sc(pn)] sc(p2)]^2
%              + [dT/dpr(pn)  w1 dT/dpr(1)   w2 dT/dpr(2)]^2 err(pref)^2 )
%  ---------------------------------------------------------------------
%
%   HOW TO CALCULATE err(pref):
%
%   THEORY:
%
%   If you have N instruments, and you want to do a fit
%   onto the canonical profile p(T) to find pref, then the least sq
%   procedure minimizes the variance between the data and the
%   modeled profile z:
%
%      0 = d/dpref[sum_i=1 ^N (pref p(i)   z(i))^2]
%  =>  pref = [sum_i=1 ^N (p(i) + z(i))] / N
%
%   So,
%    err(pref) = sqrt[ N*(err(p)/N)^2 + sum i[ (dz/dT[ i*err(T)/N) 2 ] ]
%   ---------------------------------------------------------------------
%  | err(pref) = sqrt(err(p)^2/N + (err(T)/N)^2*[sum i^N (dz(i)/dT) 2]) |
%   ---------------------------------------------------------------------
%
%   where err(T) = .2 deg and  err(p) = 5.7 db
%   and N = number of CMs on the mooring for that particular day
%
%   THE HARD PART IS KNOWING  DZ/DT  TO FIND DZ/DT, FIRST FIND Z(T)
%   PRIOR TO RUNNING TEMPCOR.M, USING PTPRO.M AND PTFIT.M. THEY FIND
%   the coeficients (H, Tmid, Tr, P0) of the functional form
%
%    z =  pref-p = H*atanh((T-Tmid)/Tr) + P0
%
%   by doing a least squares regression on all the (z,T) data
%   This functional form should be approximately equal to the
%   inverse of the canonical temperature profile  T(z)  Therefore,
%   pref of this program will be approximately (but not exactly) equal
%   to the pref of the mooring motion routines
%
%   Thus,
%    ----------------------
%   |  dz/dT = (H/Tr)*(1/(1   arg^2)),
%   |
%   |                for   1<arg<1   where  arg = (T Tmid)/Tr
%   |
%   |      arg <= -1  should be replaced by arg = -0.99
%   | and  arg >= +1  should be replaced by arg = +0.99
%    ----------------------
```

```
% NOTE: if iv = [2 3], delp = fuldelp(iv),   <<< delp is global !!
%
%              pu =  p2 - p(iv(1)) = pl+delp(1)  where delp(1)=fuldelp(2)
%              pl -  p3 - p(iv(2)) = pl+delp(2)        delp(2)=fuldelp(3)
%          pl-pu = p3-p2 =p(iv(2))-p(iv(1)) = delp(2) - delp(1);
%
%   GLOBAL VARIABLES: terlkup, cn, Cp, delp
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%function [tc,ter]=tempcor(tdr(iv),tsim(iv),tpnom,pl,pnom(inst),z);
function [tc,ter] = tempcor(tdr,tsim,tpnom,pl,pnom,z);

noerr = 0;   % if don't want to compute ter then let noerr=1

per = .05;
dt = .03;

    [nl,ncm] = size(tdr);
    np = length(pnom);
    if(np == 1)
        pnom = pnom*ones(1,nl);
    end

%  FIND WHICH 2 CM ARE THE CLOSEST AND USE THEM TO CORRECT TO PNOM
%  AND FIND ERR(TC)

if(ncm == 1)
   error('case e or case g should NOT call tempcor!')
elseif(ncm == 2),
   nl2 = 1:nl,
elseif(ncm == 3)
   nl2 = find(pnom < pl + delp(2));
   n23 = find(pnom >= pl + delp(2));
end

wl = zeros(1:nl); w2 = zeros(1:nl); pu = zeros(1:nl), pl = zeros(1:nl),
tcu = zeros(1:nl)'; tcl = zeros(1:nl)'; tc = zeros(1:nl)';
chit = zeros(1:nl);  chisc = zeros(1:nl); chipr = zeros(1:nl),

if(~isempty(nl2));
  tcu(nl2) = tdr(nl2,1) + tpnom(nl2) - tsim(nl2,1),
  tcl(nl2) = tdr(nl2,2) + tpnom(nl2) - tsim(nl2,2),
  pu(nl2) = pl(nl2) + delp(1);
  pl(nl2) = pl(nl2) + delp(2);
end

if(~isempty(n23)),
  tcu(n23) = tdr(n23,2) + tpnom(n23) - tsim(n23,2),
  tcl(n23) = tdr(n23,3) + tpnom(n23) - tsim(n23,3);
  pu(n23) = pl(n23) + delp(2);
  pl(n23) = pl(n23) + delp(3),
end

 wl = abs(pl - pnom) ./(abs(pnom-pu) + abs(pnom-pl)),
 w2 = abs(pnom - pu) ./(abs(pnom-pu) + abs(pnom-pl)),
 tc = wl' .*tcu + w2' .*tcl;   %<<<<<<<<<<<<<<CORRECTED TEMPERATURE

  nz = find(tc==0),
  if(~isempty(nz)),
     error('nl2 + n23 does not = nl in tempcor.m')
  end

if(noerr==0)    % if noerr==0 then compute error in Tcor
chit = (wl.^2 + w2.^2)*dt,
```



```
w = zeros(pu),
np = find(pnom < pu),
nm = find(pnom >= pu),
w(np)= wl(np)./(abs(pnom(np)-pu(np)) + abs(pnom(np)-pl(np))),
w(nm)= wl(nm)./(abs(pnom(nm)-pu(nm)) + abs(pnom(nm)-pl(nm))),
chipl = ((tcl'-tcu').*w.*per).^2,

np = find(pnom <= pl),
nm = find(pnom > pl);
w(np)= w2(np)./(abs(pnom(np)-pu(np)) + abs(pnom(np)-pl(np))),
w(nm)= w2(nm)./(abs(pnom(nm)-pu(nm)) + abs(pnom(nm)-pl(nm))),
chip2 = ((tcu'-tcl').*w.*per).^2,
 zpn = z + pl - pnom,
 zpu = z + pl - pu,
 zpl = z + pl - pl,

nz = find(zpl < min(terlkup(:,1)) | zpu > max(terlkup(:,1))),
nlz = find(zpl >= min(terlkup(:,1)) & zpu <= max(terlkup(:,1))),

tc(nz) = tpnom(nz),    %profile isn't defined for pref pCMl or pref pCMu
                       %so use canonical profile at pnom


scl = tablel(terlkup,zpu(nlz)),    % these are column vectors
sc2 = tablel(terlkup,zpl(nlz)),
scp = tablel(terlkup,zpn),
chis=(wl(nlz).^2).*(scp(nlz)' - scl').^2 + (w2(nlz).^2).*(scp(nlz)' - sc2').^2,

%    NOTE:
%        chip = [dTdp(pn) - wl*dTdp(1) - w2*dTdp(2)]^2 * er(pref)^2
%
% FIRST: Compute err(pref)
% err(pref)=sqrt((err(P)/ncm)^2 + (err(T)/ncm)^2*[sum [i] [ncm](dz(i)/dT)^2 ])
%
%    STEP 1A: Compute mdzdt = sum [i] [ncm] (dz(i)/dT)^2
%
%     |    dz/dT = (h/Tr)*(1/(1 - arg^2)),                      |
%     |                                                          |
%     |                 for: -1<arg<1   where: arg = (T-Tmid)/Tr |
%     |                                                          |
%
    arg = zeros(length(z),ncm),    dzdt = zeros(length(z),3),
    arg = (tdr - Cp(2))./Cp(3),
    is = find(arg<=-.95),
    arg(is) = -.95*ones(is),              %  require  -1 < arg < 1
    is = find(arg)=.99),
    arg(is) =  .99*ones(is),
    dzdt = zeros(nl,ncm),                 % dzdt is same size as tdr
    dzdt = (Cp(1)/Cp(3))./(1 - arg.^2),
if(ncm==2)
   mdzdt = dzdt(:,1).^2 + dzdt(:,2).^2,
else
   mdzdt = dzdt(:,1).^2 + dzdt(:,2).^2 + dzdt(:,3).^2,
end
    prefer = sqrt((per^2)/ncm + ((dt/ncm).^2).*mdzdt),     %column vector

%
%  NOW COMPUTE dTdz at pu pl and pn
%


 ct = [length(cn):-1:1].*cn,
 dtdpn = polyval(ct,zpn),         % these are all row vectors
 dtdpu = polyval(ct,zpu(nlz)),
```

```
dtdpl = polyval(ct,zpl(nlz));

chipr = ((dtdpn(nlz) - w1(nlz).*dtdpu - w2(nlz).*dtdpl).*prefer(nlz)').^2;

ter(nlz) = sqrt(chit(nlz)' + chis' + chipr' + chip1(nlz)' + chip2(nlz)');

ter(nz) = sqrt(scp(nz).^2 + (dtdpn(nz)'.*prefer(nz)).^2);

else                    %
 ter = zeros(z');       %
end;                    % end if noerr==0 compute ter;
```

```
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%
%                                TPREFER.M
%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%
%  The error in the mooring motion corrected temperature is:
%
%  ---------------------------------------------------------------------
%  | err(T) = sqrt( scatter(@pref-p)^2 + (dT/dpref*err(pref))^2 )      |
%  ---------------------------------------------------------------------
%
%  The scatter can be found from a lookup table (terlkup) that is
%  generated from tenv.m. The scatter depends only on pref-pnom.
%  THIS SHOULD BE DONE IN THE DRIVER PROGRAM!!  THIS PROGRAM
%  OUTPUTS:
%           tper = (dT/dpref)*err(pref);
%
%  dT/dpref depends on the canonical profile as does err(pref).
%  The Scatter, dT/dp and err(pref) are all evaluated at p=pnom,
%  however, err(pref) also depends on the errors in the (T,P)
%  pairs (and how many pairs) used to compute pref. That is,
%  err(pref) = fn((p1,p2,p3),(T1,T2,T3), err(p), err(T)).
%
%  HOW TO CALCULATE err(pref):
%
%  THEORY:
%
%  If you have N instruments, and you want to do a fit
%  onto the canonical profile p(T) to find pref, then the least sq.
%  procedure is to minimize the variance between the data and the
%  modeled profile z:
%
%     0 = d/dpref[sum_i=1 ^N (pref-p(i) - z(i))^2]
%  => pref = [sum_i=1 ^N (p(i) + z(i))] / N
%
%  So,
%   err(pref) = sqrt[ N*(err(p)/N)^2 + sum_i[ (dz/dT|_i*err(T)/N)^2 ] ]
%
%  ----------------------------------------------------------------------
%  | err(pref) = sqrt(err(p)^2/N + (err(T)/N)^2*[sum_i^N (dz(i)/dT)^2]) |
%  ----------------------------------------------------------------------
%
%  where err(T) = .2 deg and  err(p) = 3 db (= .03d) <<<<MEGHAN CHECK THIS
%  and N = number of CMs on the mooring for that particular day.
%
%  THE HARD PART IS KNOWING  DZ/DT. TO FIND DZ/DT, FIRST FIND Z(T)
%  PRIOR TO RUNNING TEMPER.M, USING PTPRO.M AND PTFIT.M. THEY FIND
%  the coeficients (H, Tmid, Tr, P0) of the functional form:
%
%   z =  pref-p = H*atanh((T-Tmid)/Tr) + P0
%
%  by doing a least squares regression on all the (z,T) data
%  This functional form should be approximately equal to the
%  inverse of the canonical temperature profile: T(z)  Therefore,
%  pref of this program will be approximately (but not exactly) equal
%  to the pref of the mooring motion routines.
%
%  Thus,
%
%   -------------------------------------------------------
%   |  dz/dT = (H/Tr)*(1/(1 - arg^2));
%   |
%   |          for  |arg|<1  where  arg = (T-Tmid)/Tr
```



```
%  |                                                          |
%  |     arg <= -1  should be replaced by arg = -0.99        |
%  | and arg >= +1  should be replaced by arg = +0.99        |
%  ------------------------------------------------------------
%
%
%  This program, temper.m should be called in the driver program
%  of the mooring motion correction after correcting the temperature.
%  It can compute the daily temp error in one fell swoop
%
%
%
%  INPUTS:      tdr   = [tcm1 tcm2 tcm3] (bad data flagged with zeros)
%               z     = daily pref pl  (note: bad values=0 so must use tcm
%               pl    = pl                       to flag when error is n/a)
%               pnom  = pnom
%             fuldelp = [0 (p2-p1) (p3-p1)]
%
%  OUTPUT:      tper  = daily error in the corrected temperature due to an
%                       error in the reference pressure.
%               prefer = daily error in the reference pressure [optional output]
%
%  GLOBAL VARIABLES:  terlkup cn Cp      (from north mat or mid mat)
%                      Cp   = [H Tmid Tr P0]  (from north mat or mid mat)
%
%                                       Meghan   Dec 24, 1991
%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

%function [tper,prefer] = tprefer(tdr,z,pl,pnom,fuldelp);
function [tper,prefer] = tprefer(tdr,z,pl,pnom,fuldelp);

  ter = .03   % error in temp measurement in degrees Celsius
  per = .07   % error in pressure measurement in db/100
%
% STEP 1: Compute terr due to scatter of (T,p) around T-z profile
%         Lookup table is pref-p vs. terr scat in 25 meter bins
%         IMPORTANT: each terr scat is good for:
%                    pref-p <= z+pl-pnom < pref-p+25
%
%
  tersc = 10*ones(1,length(zp))';
  tersc = table1(terlkup,zp);       % this makes a smooth terr
  if (~isempty(find(tersc==10)))
    disp('there are some zp values that are out of the lookup table range')
  end
%
%
% STEP 2. Compute err(pref)
  err(pref) = sqrt((err(P)/N)^2 + (err(T)/N)^2*[sum (i) (N) (dz(i)/dT)^2 ])
%
  STEP 2A. Compute mdzdt = sum (i) (N) (dz(i)/dT)^2
%
% |  dz/dT = (H/Tr)*(1/(1 - arg^2)),                        |
% |                                                          |
% |            for  |arg|<1   where  arg = (T-Tmid)/Tr       |
% ------------------------------------------------------------
%
  arg = zeros(length(z),1);    dzdt = zeros(length(z),1);
  arg = (tdr - Cp(2)) /Cp(3);
  is = find(arg<=-.95);
  arg(is) = -.95*ones(is);              % require  -1 < arg < 1
  is = find(arg>=.99);
  arg(is) = .99*ones(is);
```

```
    dzdt = (Cp(1)/Cp(3))./(1 - arg.^2);
%                                                          Have CMs:
    ia = find(tdr(:,1)~=0 & tdr(:,2)~=0 & tdr(:,3)~=0);   % a: 1 2 3
    ib = find(tdr(:,1)~=0 & tdr(:,2)==0 & tdr(:,3)~=0);   % b: 1   3
    ic = find(tdr(:,1)~=0 & tdr(:,2)~=0 & tdr(:,3)==0);   % c: 1 2
    id = find(tdr(:,1)==0 & tdr(:,2)~=0 & tdr(:,3)~=0);   % d:   2 3
    ie = find(tdr(:,1)~=0 & tdr(:,2)==0 & tdr(:,3)==0);   % e: 1
    ig = find(tdr(:,1)==0 & tdr(:,2)==0 & tdr(:,3)~=0);   % g:     3

    mdzdt = zeros(length(z),1);
    mdzdt(ia) = dzdt(ia,1).^2 + dzdt(ia,2).^2 + dzdt(ia,3).^2;
    mdzdt(ib) = dzdt(ib,1).^2 + dzdt(ib,3).^2;
    mdzdt(ic) = dzdt(ic,1).^2 + dzdt(ic,2).^2;
    mdzdt(id) = dzdt(id,2).^2 + dzdt(id,3).^2;
    mdzdt(ie) = dzdt(ie,1).^2;
    mdzdt(ig) = dzdt(ig,3).^2;
%
%  STEP 2B: compute prefer = err(pref)
%
   prefer = sqrt((per^2)/2 + ((ter/2)^2)*mdzdt);
if(~isempty(ia))
   prefer(ia) = sqrt(ones(ia)*((per^2)/3) + ((ter/3)^2)*mdzdt(ia));
end
if(~isempty(ie))
   prefer(ie) = sqrt(ones(ie)*(per^2) + (ter^2)*mdzdt(ie));
end
if(~isempty(ig))
   prefer(ig) = sqrt(ones(ig)*(per^2) + (ter^2)*mdzdt(ig));
end

%
%  STEP 3: Compute tper
%
 zp = z'*pl'-pnom';
 ct = [length(cn):-1:1].*cn;
 dtdp = polyval(ct,zp);
 tper = (dtdp.*prefer);
%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
```

```
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%
%                                VELC.M
%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%function [U,errU] = velc(tc, tdr(nl,iv(:)), u(nl,iv(:)), v(nl,iv(:)),ter(nl))
%          This .m function corrects the downstream (shear) component
%          of the velocity to the nominal pressure by interpolating
%          on the temperature.
%
%          case = [1 2 3] for case a
%                 [1 3] for b etc
%  INPUT:
%          tc = corrected temp time series
%          u = u(nl,case)
%          v = v(nl,case);
%          tr= tr(nl,case);  observed and simulated (case is determined by v)
%          ter = error in tc
%
%  OUTPUT:
%          U = [uc,vc]  east and north components.
%          errU=[uer,ver] error in corrected components
%
%  NOTE: in code:
%          delu = udo_cor - udo_lev;  in downstream direction. Use for error
%          m  =  (udo_1 - udo_2)./(tr(1) - tr(2));   for error analysis.
%          theta = angle of velocity shear time series.
%
% function [U,errU] = velc(tc,tdrall,uall,vall,ter);
 function [U,errU] = velc(tc,tdrall,uall,vall,ter);

[nl,nom] = size(tdrall);

% Sort inst by how close to poom (daily)
 if(nom == 2)
     [dum,ii] = sort(abs([tdrall(:,1)-tc tdrall(:,2)-tc]'));
     livl = ii(1,:)';
     liv2 = ii(2,:)';    % 2nd closest to poom.  This is index of next closests
     n12 = find(livl == 1 & liv2 == 2);
     n21 = find(livl == 2 & liv2 == 1);
 else
     [dum,ii] = sort(abs([tdrall(:,1)-tc tdrall(:,2)-tc tdrall(:,3)-tc]'));
     livl = ii(1,:)';
     liv2 = ii(2,:)';    % 2nd closest to poom.  This is index of next closests
     n12 = find(livl == 1 & liv2 == 2);
     n21 = find(livl == 2 & liv2 == 1);
     n23 = find(livl == 2 & liv2 == 3);
     n32 = find(livl == 3 & liv2 == 2);
end
     u = 999*ones(nl,2);

if(~isempty(n12))
     u(n12,:) = uall(n12,[1 2]);% inst closest to poom
     v(n12,:) = vall(n12,[1 2]);
     tr(n12,:)= tdrall(n12,[1 2]);
end
if(~isempty(n21))
     u(n21,:) = uall(n21,[2 1]);
     v(n21,:) = vall(n21,[2 1]);
     tr(n21,:) = tdrall(n21,[2 1]);
end
if(~isempty(n23))
     u(n23,:) = uall(n23,[2 3]);
     v(n23,:) = vall(n23,[2 3]);
```



```
     tr(n23,:) = tdrall(n23,[2 3]);
end
if(~isempty(n32))
     u(n32,:) = uall(n32,[3 2]);
     v(n32,:) = vall(n32,[3 2]);
     tr(n32,:) = tdrall(n32,[3 2]);
end
if( ~isempty(find(u == 999)))
     error('not sorting velocities and temps right in velc.m')
end

      theta = atan2(v(:,1)-v(:,2),u(:,1)-u(:,2));

      ur(:,1)=u(:,1).*cos(theta) + v(:,1).*sin(theta);
      ur(:,2)=u(:,2).*cos(theta)+v(:,2).*sin(theta);
      vr(:,1)=-u(:,1).*sin(theta)+v(:,1).*cos(theta);
      sc = ( ur(:,1).*( tc-tr(:,2) ) + ur(:,2).*( tr(:,1)-tc ) );
      sc = sc./( tr(:,1) - tr(:,2) );
      uc=sc.*cos(theta)-vr(:,1).*sin(theta);
      vc=sc.*sin(theta)+vr(:,1).*cos(theta);
       U = [uc vc];

%NOW CALCULATE ERROR IN U


% extrapolation when:
   nex = find(abs(tc-tr(:,2)) > abs(tr(:,1)-tr(:,2)));
   dm = .01*ones(length(tc),1);
   dm(nex) = .02*ones(length(nex),1);

   du = .02;
   dth = 5*pi/180;
   ter = ter;

   delu = sc - ur(:,1);
   m = (ur(:,1) - ur(:,2))./(tr(:,1) - tr(:,2));

uchith = (delu.*sin(theta).*dth).^2;
uchit = (m.*cos(theta).*ter).^2;
uchim = ((tc-tr(:,1)).*cos(theta).*dm).^2;
uer = sqrt(uchith + uchit + uchim + du^2);

vchith = (delu.*cos(theta).*dth).^2;
vchit = (m.*sin(theta).*ter).^2;
vchim = ((tc-tr(:,1)).*sin(theta).*dm).^2;
ver = sqrt(vchith + vchit + vchim + du^2);

errU = [uer ver];
```

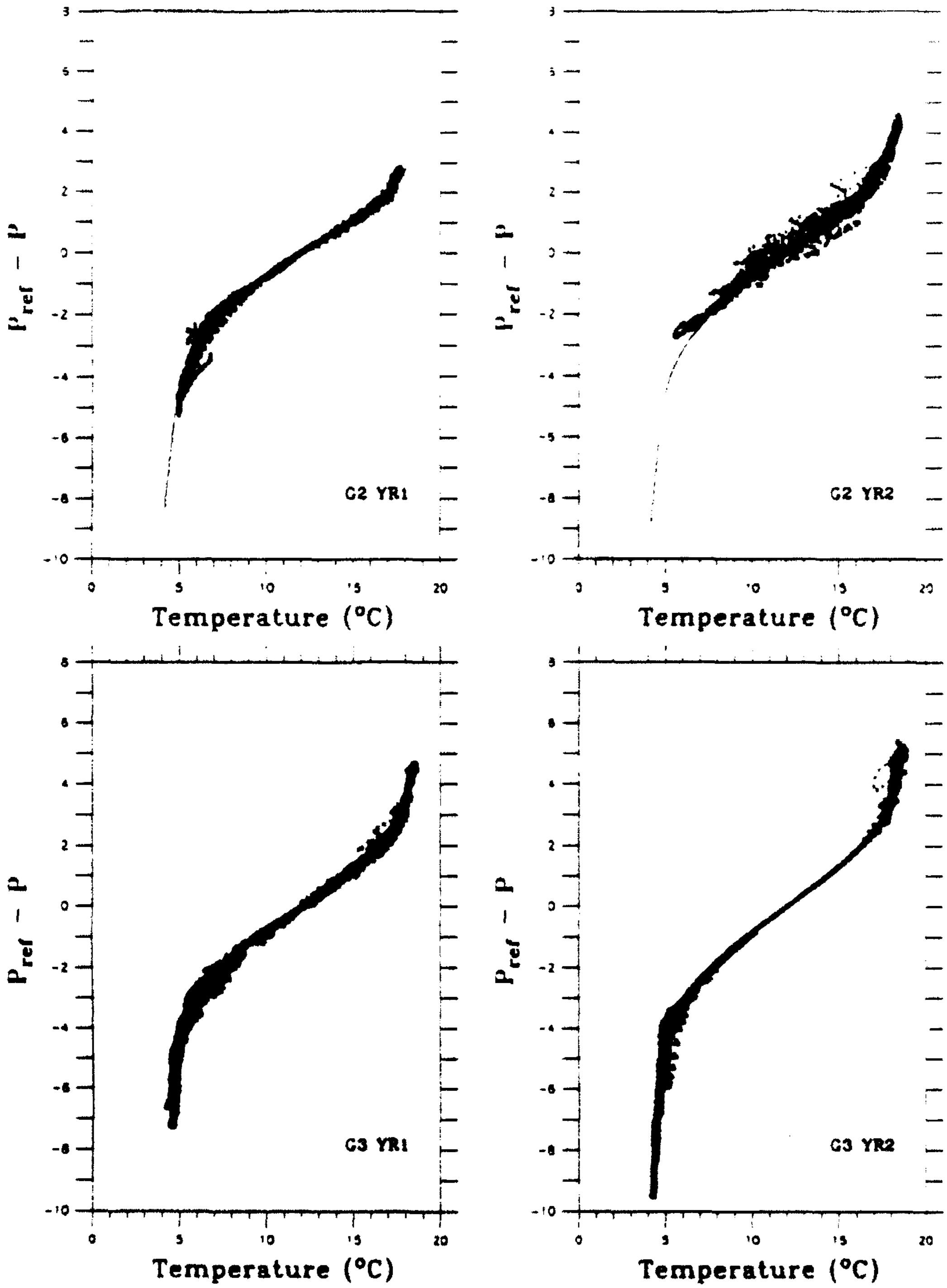
```
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%
%                              TENV.M
%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%
%   tenv.m computes a look up table for the std envelope of the
%   canonical temp profile as a function of pref-p.
%   (NOTE:  pref-p = (pref-pl) - (p-pl) = z - delp)
%   where pref-p are grouped in bins of 25 meters.
%
%   This program should be run after synopbz.m and before
%   mmcorn.m or mmcors.m, i.e. after (or while) computing the
%   canonical profile, but before using this profile to correct
%   the temperature.
%
%   When calculating the error in the corrected temp, this lookup
%   table will be used along with a program which will estimate
%   the error in T due to an error in the reference pressure.
%
%   INPUTS:  t = temperature array used to create the canonical profile
%            zsyn = pref - pl corresponding to t, (output of synopbz.m)
%            delp = pressure offsets of t(1), t(2) and t(3) from t(1).
%              cn = polynomial coeficients of canonical profile.
%
%   OUTPUTS: npts = vector of number of points used in each bin.
%          terrlkup = [p tproerr] = pressure & std envelope for t profile.
%                      NOTE: p is in middle of bin p. For example,
%                      tproerr(1) is appropriate for p(1)-10 <= p < p(2)+10
%                      where p(2) = p(1) + 20.
%
%                                      meghan  12/16/91
%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%
% function [terlkup, npts] = tenv(tsyn,zsyn,delp,cn);
function [terlkup, npts] = tenv(tsyn,zsyn,delp,cn);

% STEP 1:  string t and z-delp into one vector

i1 = find(tsyn(:,1)~=0);
i2 = find(tsyn(:,2)~=0);
i3 = find(tsyn(:,3)~=0);
t = [tsyn(i1,1); tsyn(i2,2); tsyn(i3,3)];
z = [zsyn(i1)'; zsyn(i2)'-delp(2); zsyn(i3)'-delp(3)];

% STEP 2: sort z into ascending order keeping track of indices.

[zs, i] = sort(z);
ts = t(i);

% STEP 3: figure out how many bins

p = (zs(1)+.1:.20:max(zs))';
k = find(zs>=p(1)-.1 & zs<p(1)+.1);
npts = length(k);
while npts<2;
  zs = zs(2:length(zs));
  ts = ts(2:length(ts));
  p = (min(zs)+.1:.2:max(zs)-.1)';
  k = find(zs>=p(1)-.1 & zs<p(1)+.1);
  npts = length(k);
end
```



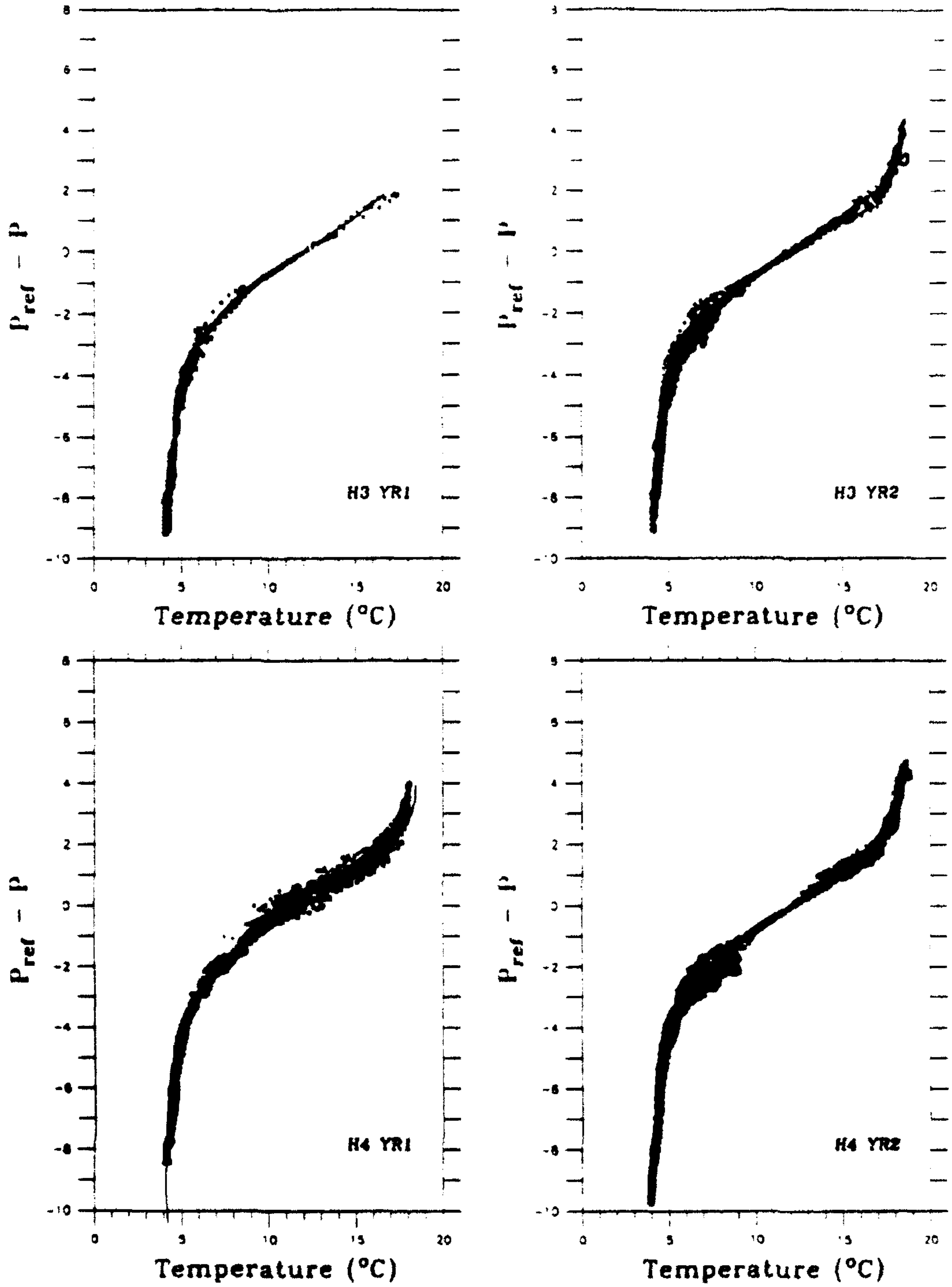
```
nb = length(p);

npts = [];
terlkup = [];
for n = 1:nb
n
 k = find(zs>=p(n)-.1 & zs<p(n)+.1)
if(isempty(k))
 fill = npts(n-1);
 npts = [npts;fill];
 fill = terlkup(n-1,:);
 terlkup = [terlkup;fill];
else
 npts = [npts;length(k)];
 tpro = polyval([cn 0],zs(k)) + 12;
 er = tpro - ts(k);
 terlkup = [terlkup; [p(n) sqrt(er'*er/length(er))]];
end % if no points in bin use last bin
end % bin = n

% STEP 4: PLOT

 tpro = polyval([cn 0],p) + 12;
 tper = tpro + terlkup(:,2);
 tmer = tpro - terlkup(:,2);
plot(ts,zs,'x',tpro,p,tper,p,tmer,p),title('canonical profile with std envelope'
```

Appendix C: Temperature versus Pressure Profiles

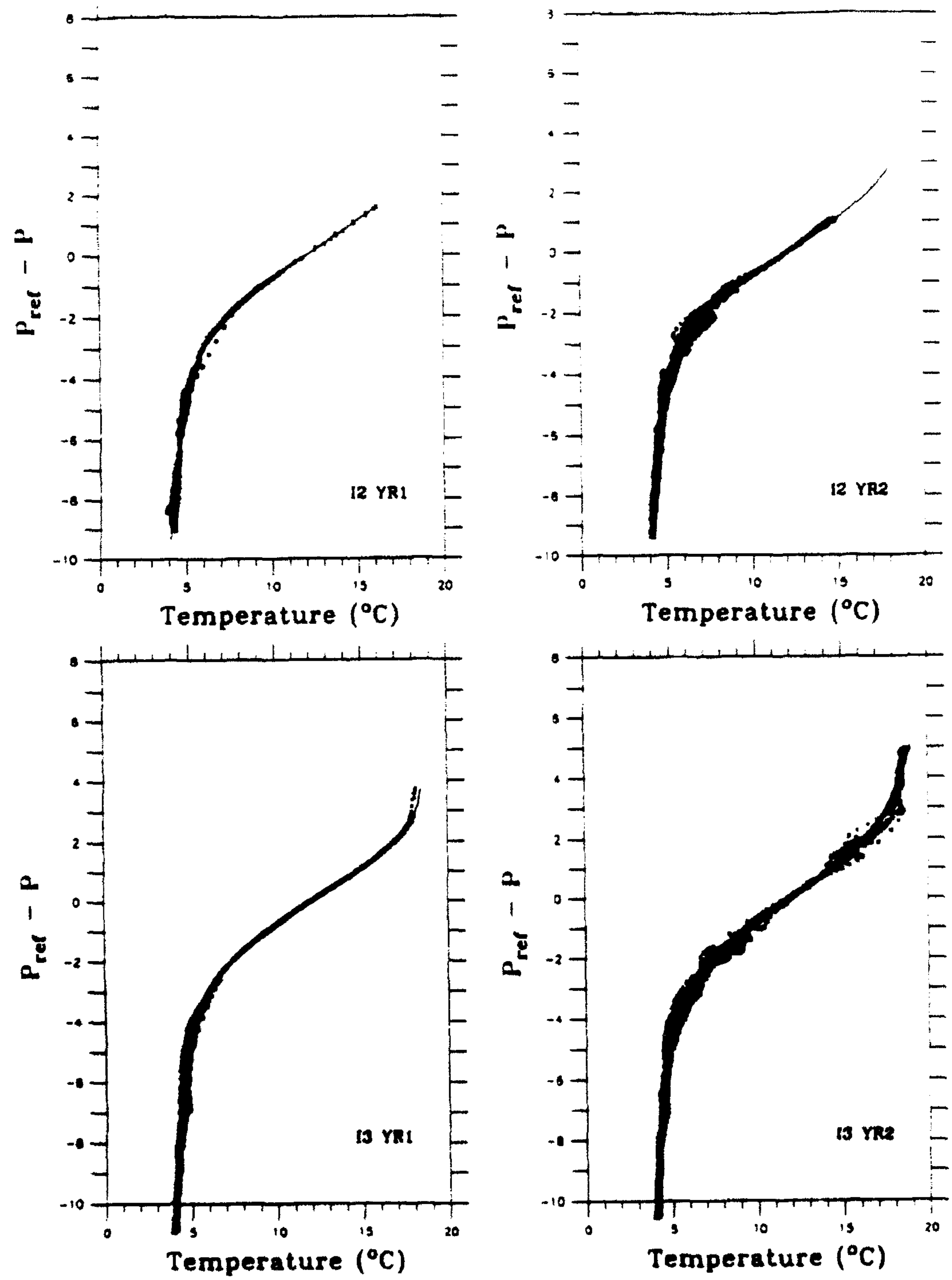
Measured temperatures are plotted against the pressure, $p_{ref} - p$, for each mooring. Level 1 data are indicated by crosses, level 2 data by squares, and level 3 data by triangles. The canonical profile is also shown for each site. The reference pressure, p_{ref}, was determined by least-squares regression for all sites except two. For sites G2_YR2 and H4_YR1, p_{ref} was obtained from the IES data.

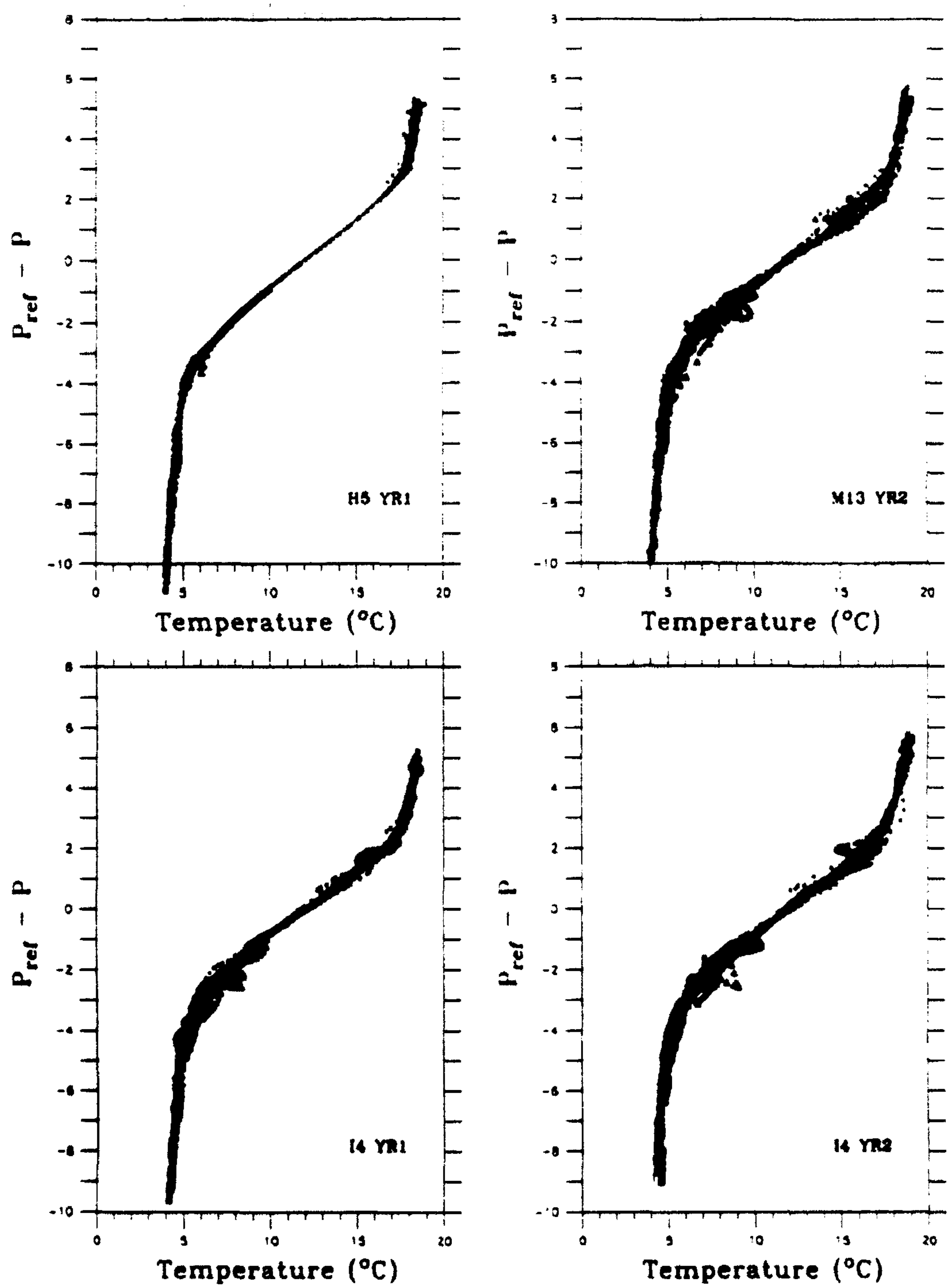


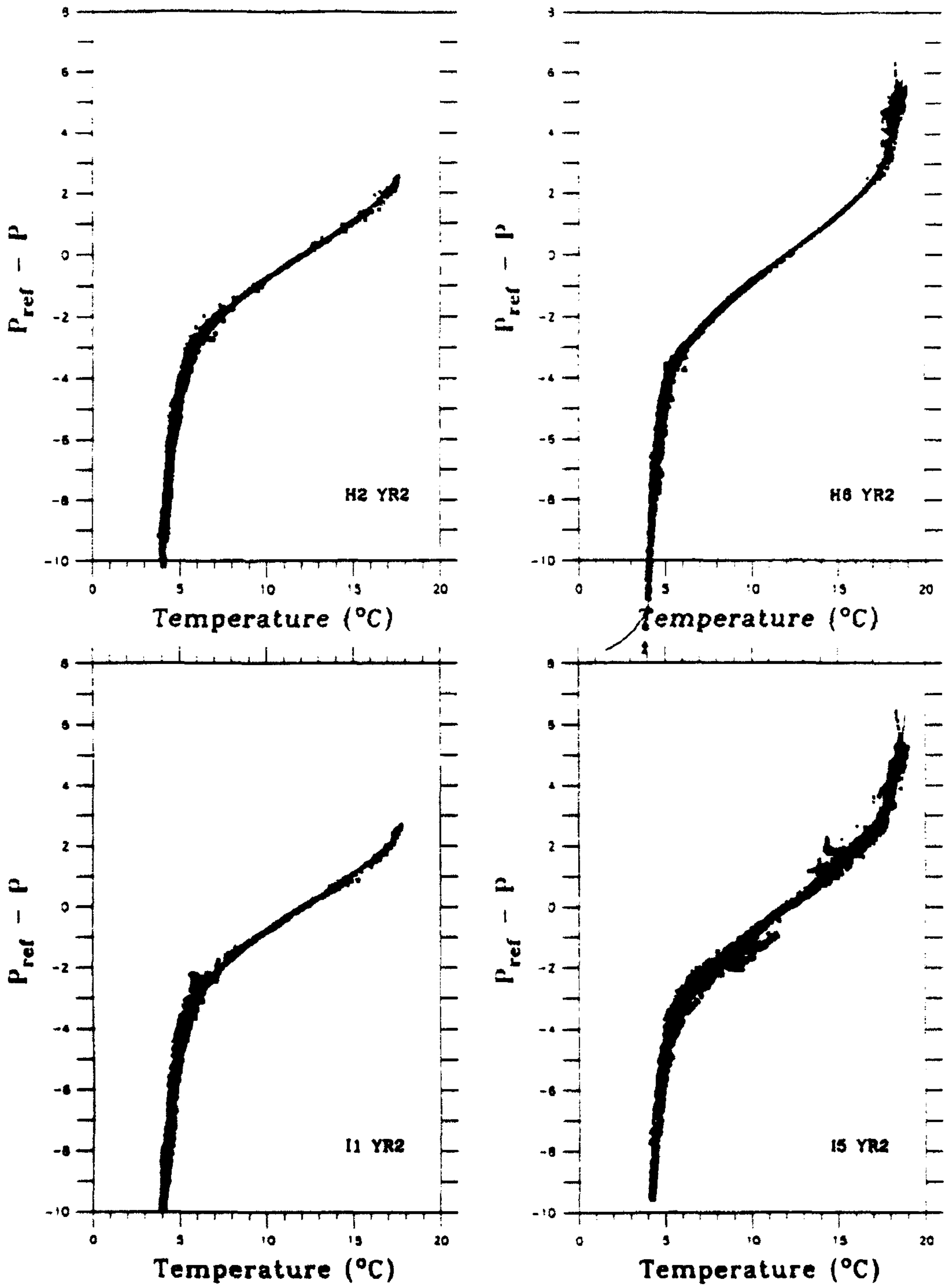
G2 YR1
G2 YR2
G3 YR1
G3 YR2
Temperature (°C)
$P_{ref} - P$



H3 YR1
H3 YR2
H4 YR1
H4 YR2
$P_{ref} - P$
Temperature (°C)


I2 YR1
I2 YR2
I3 YR1
I3 YR2
$P_{ref} - P$
Temperature (°C)


H5 YR1
M13 YR2
I4 YR1
I4 YR2
$P_{ref} - P$
Temperature (°C)



$P_{ref} - P$
H2 YR2
Temperature (°C)
$P_{ref} - P$
H6 YR2
Temperature (°C)
$P_{ref} - P$
I1 YR2
Temperature (°C)
$P_{ref} - P$
I5 YR2
Temperature (°C)

Appendix D: Pseudo-IES and IES Z_{12} Records

Time series of the depth 12°C isotherm as determined by the current meter moorings and the IESs are presented.

The current meter reference pressure records, p_{ref}, have been scaled by a factor of 1.01 to convert the units from decibars into meters. We refer to these scaled data as 'pseudo-IES' records.

The actual IES observations are not shown in the following figures because the IES and current meter sites were separated by as much as 5 km. Instead, we interpolated objective maps of the IES Z_{12} fields to obtain time series of Z_{12} right at the current meter mooring locations. These interpolated records are the ones presented here.

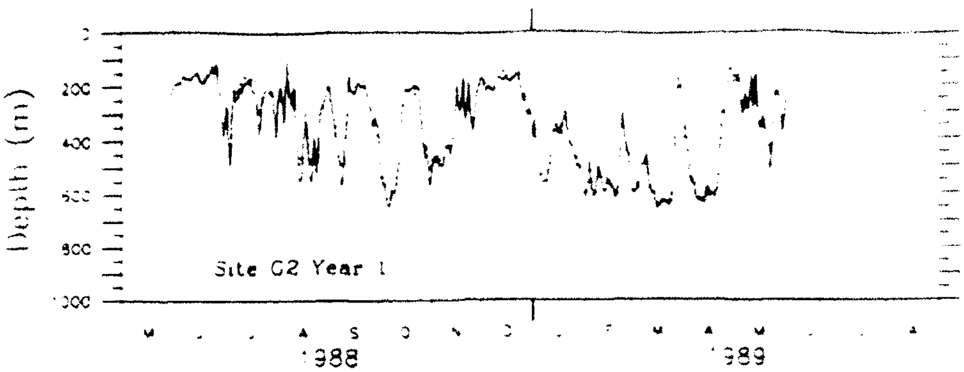
In the following figures, the IES data are shown by the dashed lines and the pseudo-IES records by the solid lines. Due to IES instrument failures, either partial data or no data are shown for the IESs at sites H2_YR1, I1_YR1, I3_YR1, and I1_YR2. We did not deploy an IES at the base of the mooring at site M13_YR2; thus no IES data is shown for that site. Additionally, no pseudo-IES data are shown for sites H4_YR1 and G2_YR2 because there was insufficient current meter data to determine p_{ref}.

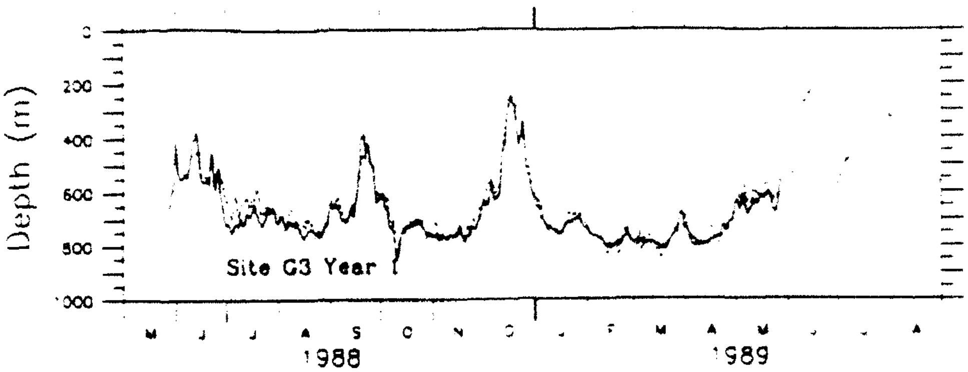
For convenience, the Year 1 data are plotted from May 1988 to August 1989 and the Year 2 data for May 1989 to August 1990. Consequently there is an overlap of approximately three months in these figures. Thus some of the current meter data (specifically, the four two-year moorings at sites H2, H6, I1, and I5) are repeated during that time period. Except for the IES sites noted above, the IES records are continuous throughout the two year period.

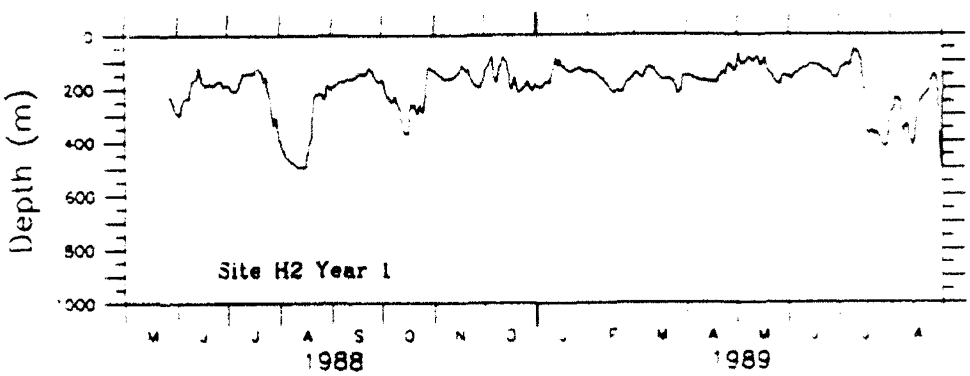
Table 9. Statistics on Pseudo_IES Z_{12} Data

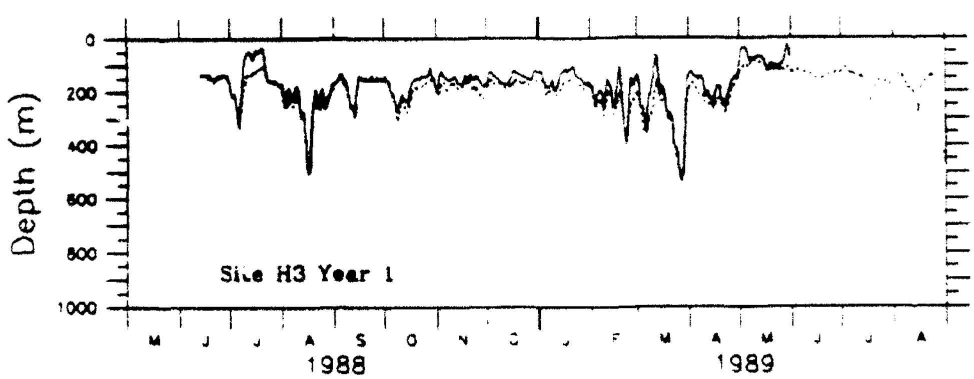
Units of Z_{12} are meters. Dividing by these values by 1.01 converts thedepths into p_{ref} in decibars. The values listed for sites H4_YR1 and G2_YR2 are the statistics of the IES, not the pseudo_IES, data because there were insufficient current meter measurements to obtain p_{ref} at those two sites. For sites H2_YR2, H6_YR2, I1_YR2, and I5_YR2, statistics are reported on the two-year-long data records (May 1988 to August 1990). For all other sites, the statistics are on data records of appproximately one year in duration.

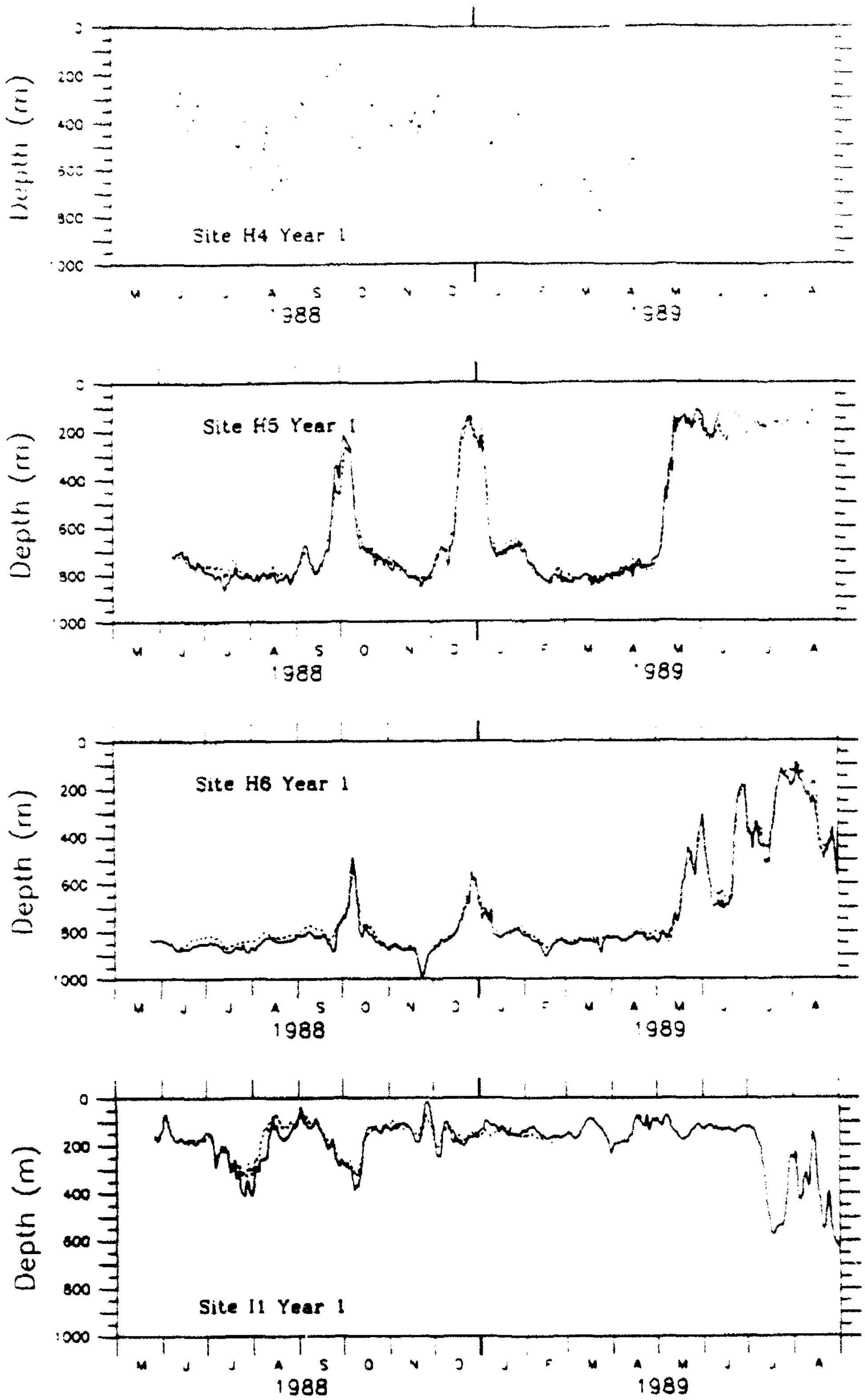
Site	Min	Max	Mean	Std
G2_YR1	102.03	658.24	355.73	158.41
G2_YR2	62.09	806.90	482.73	214.83
G3_YR1	250.68	852.07	682.56	111.56
G3_YR2	68.98	911.82	685.76	194.69
H2_YR2	-15.59	626.68	209.41	116.00
H3_YR1	22.84	536.79	172.96	77.96
H3_YR2	22.64	744.69	338.12	222.33
H4_YR1	101.29	790.25	435.49	180.21
H4_YR2	67.27	820.70	498.85	266.83
H5_YR1	113.03	867.69	663.24	220.22
H6_YR2	88.93	994.59	741.44	184.81
I1_YR2	6.71	644.99	206.34	123.83
I2_YR1	17.54	509.37	154.32	54.84
I2_YR2	16.70	744.86	326.04	216.70
I3_YR1	47.61	702.68	326.02	174.87
I3_YR2	57.47	810.38	501.09	262.75
I4_YR1	-3.82	844.03	622.74	231.74
I4_YR2	40.26	901.04	627.62	231.23
I5_YR2	65.48	999.24	714.94	200.42
M13_YR2	84.21	897.59	623.16	230.91

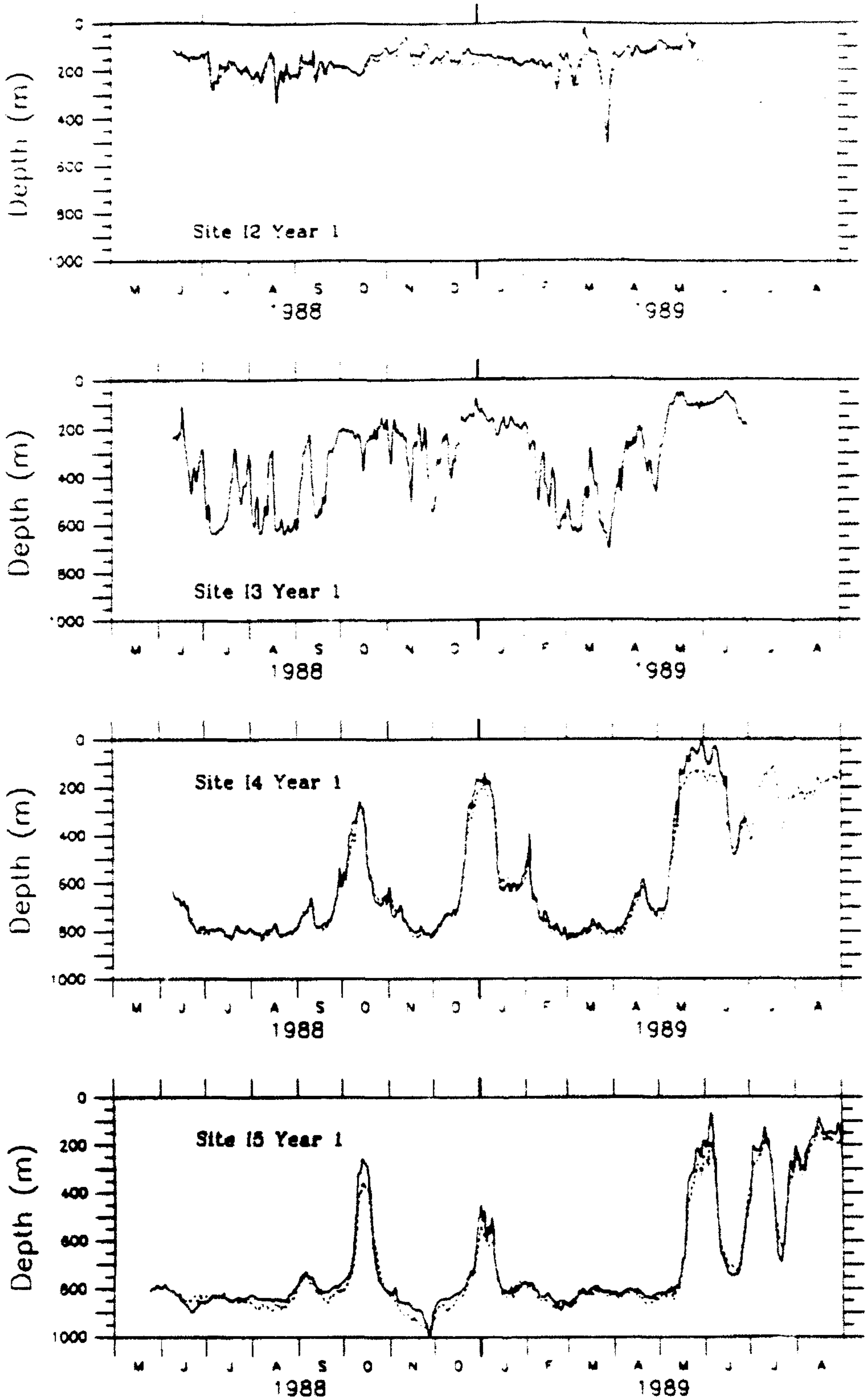

Depth (m)
Site G2 Year 1
1988
1989

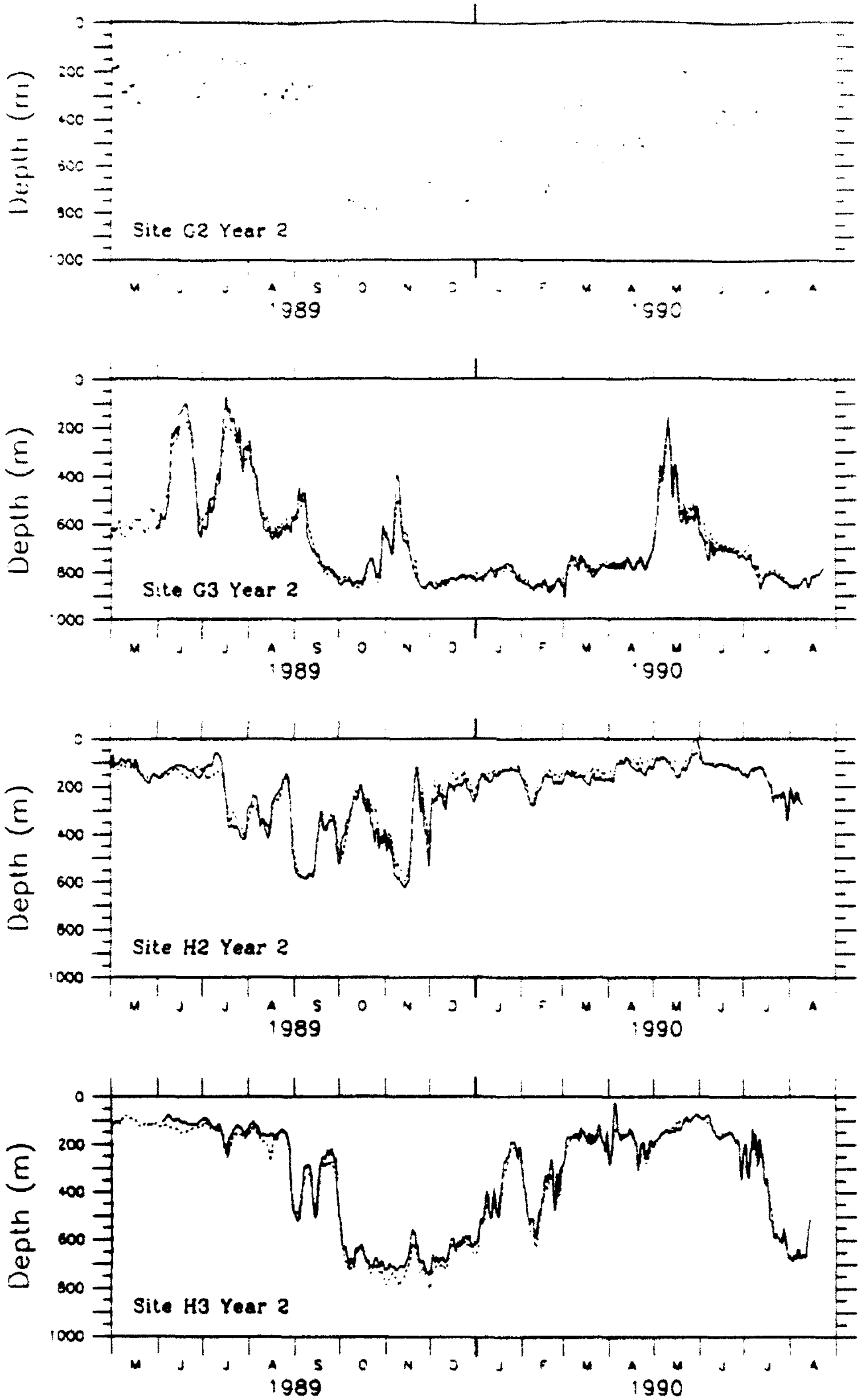

Depth (m)
Site G3 Year 1
1988
1989

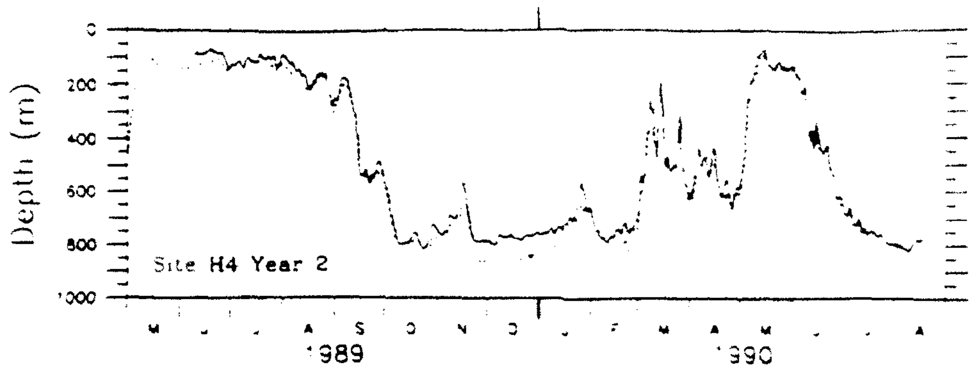

Depth (m)
Site H2 Year 1
1988
1989

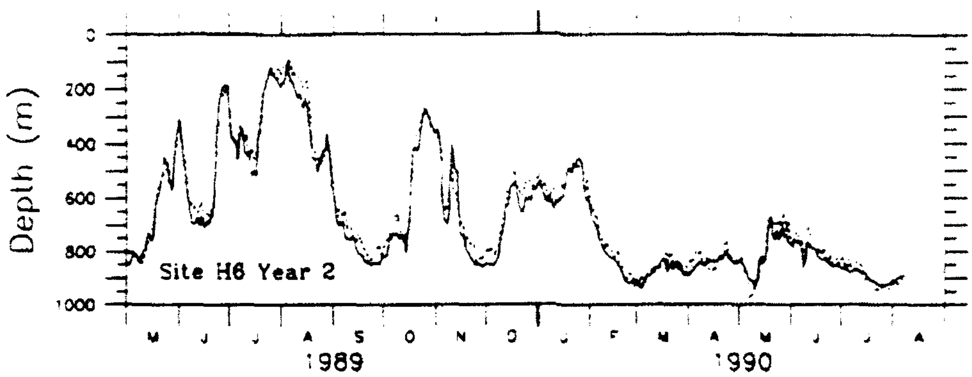

Depth (m)
Site H3 Year 1
1988
1989

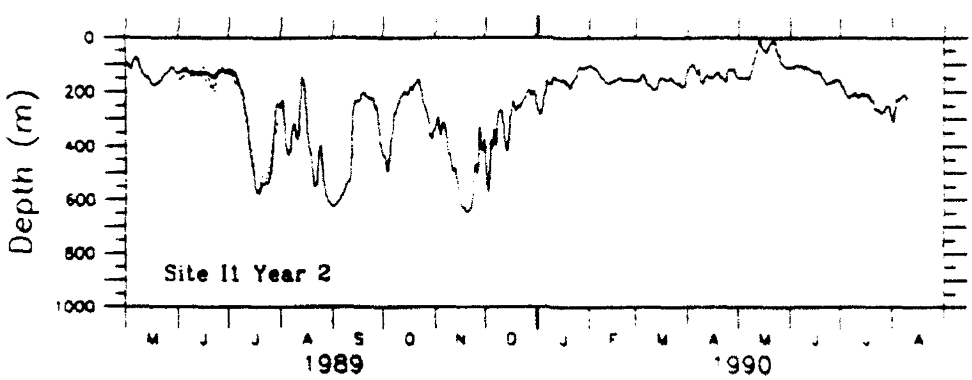

Depth (m)
Site H4 Year 1
Site H5 Year 1
Site H6 Year 1
Site I1 Year 1
1988
1989

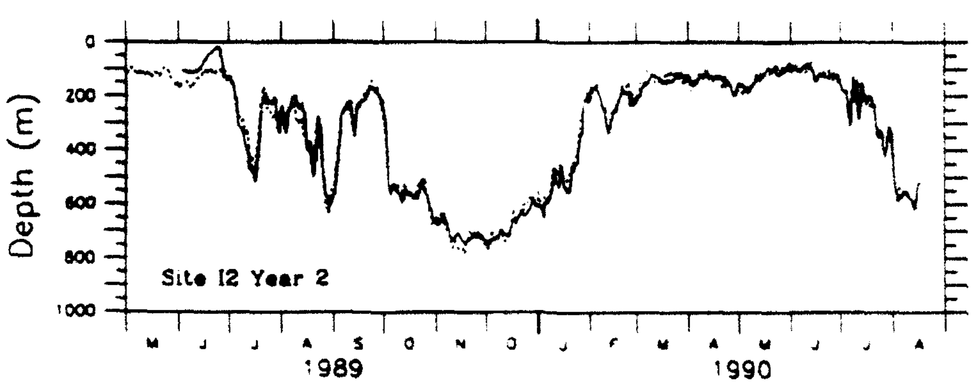

Site I2 Year 1
Site I3 Year 1
Site I4 Year 1
Site I5 Year 1
Depth (m)
0
200
400
600
800
1000
M J J A S O N D J F M A M J J A
1988
1989

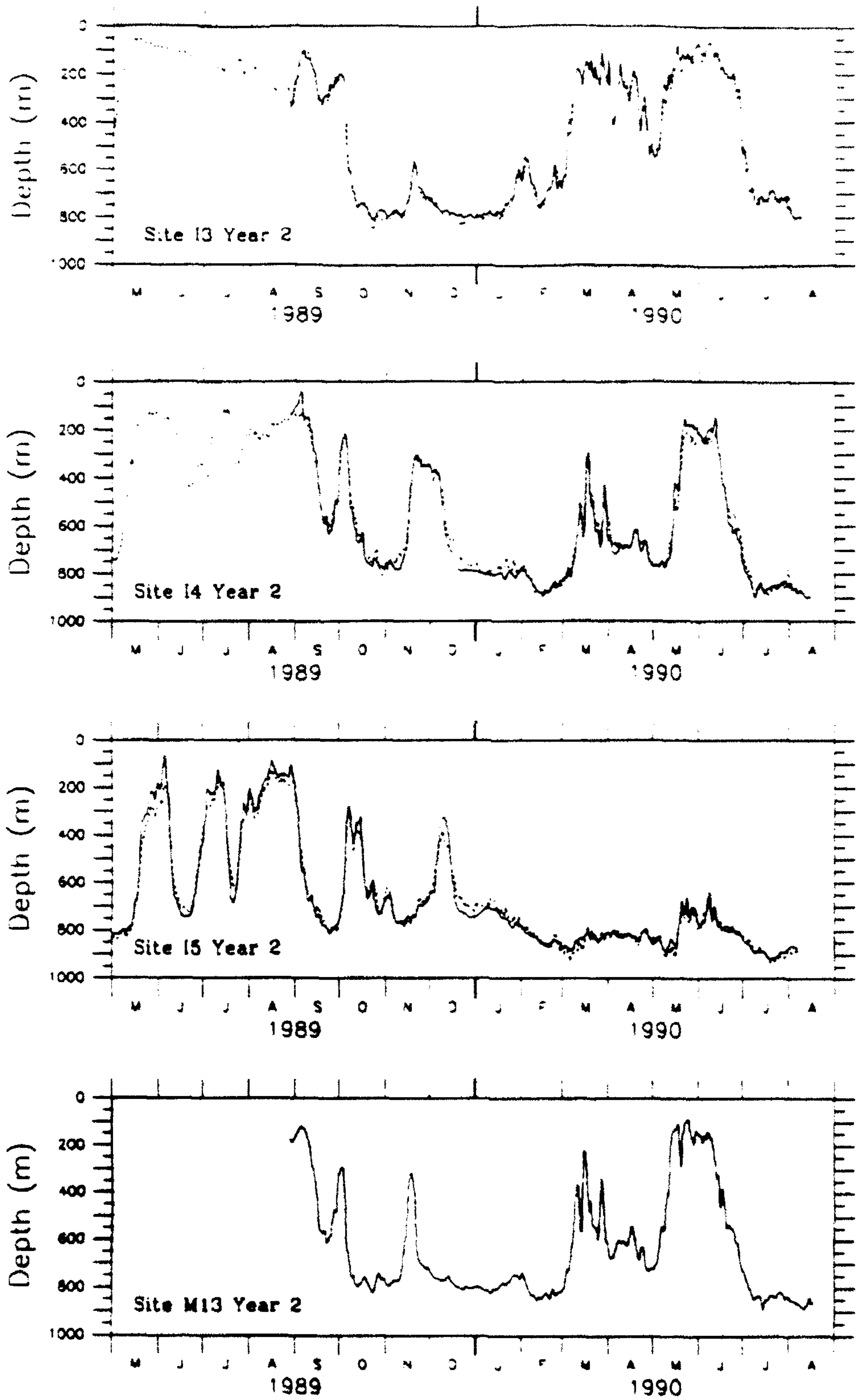

Depth (m)
Site G2 Year 2
Site G3 Year 2
Site H2 Year 2
Site H3 Year 2
0
200
400
600
800
1000
M J J A S O N D J F M A M J J A
1989
1990


Depth (m)
0
200
400
600
800
1000
Site H4 Year 2
M J J A S O N D J F M A M J J A
1989
1990


Depth (m)
0
200
400
600
800
1000
Site H6 Year 2
M J J A S O N D J F M A M J J A
1989
1990


Depth (m)
0
200
400
600
800
1000
Site I1 Year 2
M J J A S O N D J F M A M J J A
1989
1990


Depth (m)
0
200
400
600
800
1000
Site I2 Year 2
M J J A S O N D J F M A M J J A
1989
1990


Depth (m)
0
200
400
600
800
1000
Site I3 Year 2
Site I4 Year 2
Site I5 Year 2
Site M13 Year 2
M J J A S O N D J F M A M J J A
1989
1990

Appendix E: Comments on Mooring Motion Corrections

G2_YR1

- Not used to determine either profile.
- Corrected using northern profile.
- Level 4 velocities were never used due to highly-energetic events.

G3_YR1

- Not used to determine either profile.
- Corrected with southern profile.
- Special handling required to make sure all levels began at same time.
- Data gaps in level 1 velocities. Filled by extrapolation from levels 2 and 3.

H3_YR1

- Used to determine northern and southern profiles.
- Corrected using northern profile.
- Level 4 velocities were never used due to highly-energetic events.
- ADCP pressures used. However, level 1 current meter pressures were used to fill a data gap at end of record. A bias of 6 db was subtracted from the current meter pressures before using them.

H4_YR1

- Not used to determine either profile.
- Corrected with southern profile.
- ADCP pressures used instead of level 1 current meter pressures..
- Only one working temperature sensor.
- Used IES Z_{12} as p_{ref} after scaling from meters to decibars.
- A comparison of the p_{ref} (determined from the current meter data for a short time period) and the IES Z_{12} showed a 10 db bias between the two records. Since we used the IES Z_{12} data for the mooring motion correction, we added 10 db to the ADCP pressures to make the two data sets consistent.
- Biases in the current meter pressures are the sources for the discrepancies between delp12 of Table 5 and the observed delta pressures in Tables 2 and 3.

H5_YR1

- Not used to determine either profile.
- Corrected with southern profile.

I2_YR1

- Not used to determine either profile.
- Corrected using northern profile.
- Level 4 velocities were never used due to highly-energetic events.

- ADCP failed. Used level 1 current meter pressures after removing the 6 db bias.
- Special handling required to make sure all levels began at same time.

I3_YR1

- Used to determine northern profile.
- Corrected using northern profile.
- Special handling to truncate level 1 to same length as levels 2 and 3.

I4_YR1

- Used to determine southern profile.
- Corrected with southern profile.
- Special handling to truncate level 3 to same length as levels 1 and 2.

G2_YR2

- Not used to determine either profile.
- Corrected with southern profile.
- Level 4 velocities were never used due to highly-energetic events.
- Only one working temperature sensor.
- Used IES Z_{12} as p_{ref} after scaling from meters to decibars.

G3_YR2

- Not used to determine either profile.
- Corrected with southern profile.

H2_YR1 and YR2

- Two-year long record
- Used to determine northern profile.
- Corrected with northern profile.
- Level 4 velocities were never used due to highly-energetic events.
- Data gaps in level 1 velocities. Filled by extrapolation from levels 2 and 3.

H3_YR2

- Used to determine northern and southern profiles.
- Corrected using northern profile.
- ADCP pressure used.
- Level 1 current meter failed after a short period. ADCP temperatures and Bin 1 velocities were used for the remaining time period.
- Level 4 velocities were never used due to highly-energetic events.

H4_YR2

- Not used to determine either profile.

- Corrected using northern profile.
- ADCP failed. Used the level 1 current meter pressures after subtracting a 6 db bias.

H6_YR1 and YR2

- Two-year long record.
- Not used to determine either profile.
- Corrected with southern profile.
- Very large excursion of 550 m used taken by mooring.

I1_YR1 and YR2

- Two-year long record.
- Used to determine northern profile.
- Corrected using northern profile.
- Special handling to truncate last data point of all levels except level 1.
- Level 4 velocities were never used due to highly-energetic events.

I2_YR2

- Used to determine northern profile.
- Corrected using northern profile.
- ADCP pressures used.
- Level . current meter failed. Used ADCP temperatures and velocities instead.
- Level 4 velocities were never used due to highly-energetic events.

I3_YR2

- Not used to determine either profile.
- Corrected with southern profile.
- Level 2 temperatures get bad near the end of record. Use only the first 1230 data points of this record.
- Level 1 velocities failed after a short period. Thus corrected velocities at this level were obtained by extrapolation from levels 2 and 3. The largest velocities were obtained for this site (Table D1).

I4_YR2

- Used to determine southern profile.
- Corrected with southern profile.

I5_YR1 and YR2

- Two-year long record.
- Used to determine southern profile.
- Corrected with southern profile.
- Data gaps in level 1 velocities. Many gaps filled by extrapolation from levels 2 and 3. Some periods were not filled because the mooring took large excursions.

M13_YR2

- Used to determine southern profile.
- Corrected with southern profile.
- Level 1 pressures were bad. Used level 2 pressures and adjusted delp12 and delp13 accordingly (see Table 5).

Appendix F: Mooring Motion Corrected Data

Plots of the mooring motion corrected temperature and velocities are shown for each mooring. All data have been corrected to constant depths of 400 m, 700 m, and 1000 m. These data have also been lowpassed using a 40-hr Butterworth filter.

The plots on each page are organized in the same manner. In the uppermost panel, corrected temperatures at all three depth levels are shown. The corrected velocities are shown in the bottom three panels, one for each of the pressure horizons. The solid line in each panel shows the u-component of velocity and the dashed line indicates the v-component. Some data gaps are still evident in these figures.

The temperature and velocity records for each mooring are presented on two pages. The first page shows the data for the Year 1 deployment period, May 1988 to August 1989. The Year 2 data, May 1989 to August 1990, are shown on the second page. There is a three-month overlap in these two figures; thus the corrected data at several sites are repeated in the two figures.

The reference pressure, p_{ref}, at each mooring are not shown in this appendix. They are shown in Appendix D together with the corresponding IES Z_{12} records.

Table 10. Start and End Times of the Mooring Motion Corrected Data
The times are associated with the first and last points of the corrected temperature and velocity data records for all three levels on each mooring. The sampling interval is 6 hours and the record lengths are listed as the number of points.

Site	Start Time		End Time		Npts
	Date	Time (UT)	Date	Time (UT)	
G2_YR1	88-05-28	0000	89-06-03	1200	1487
G2_YR2	89-06-06	1800	90-08-24	1200	1776
G3_YR1	88-05-27	1800	89-05-27	1800	1461
G3_YR2	89-05-31	0000	90-08-24	0600	1802
H2_YR2	88-05-26	1800	90-08-10	0600	3223
H3_YR1	88-06-12	1800	89-05-31	1200	1412
H3_YR2	89-06-03	1800	90-08-14	1200	1748
H4_YR1	88-06-15	1800	89-06-05	0600	1419
H4_YR2	89-06-09	0000	90-08-18	1200	1743
H5_YR1	88-06-09	0000	89-06-14	1200	1483
H6_YR2	88-05-23	1800	90-08-08	1200	3228
I1_YR2	88-05-26	0000	90-08-10	0600	3226
I2_YR1	88-06-10	1800	89-05-29	0600	1411
I2_YR2	89-06-02	0000	90-08-16	0600	1762
I3_YR1	88-06-10	0600	89-07-01	0000	1544
I3_YR2	89-08-26	0000	90-08-09	1200	1395
I4_YR1	88-06-09	1800	89-06-28	1800	1537
I4_YR2	89-08-27	1800	90-08-15	1200	1412
I5_YR2	88-05-24	1800	90-08-07	1200	3220
M13_YR2	89-08-28	1800	90-08-17	1200	1416

Table 11. Statistics on the Mooring Motion Corrected Temperature and Velocity Data

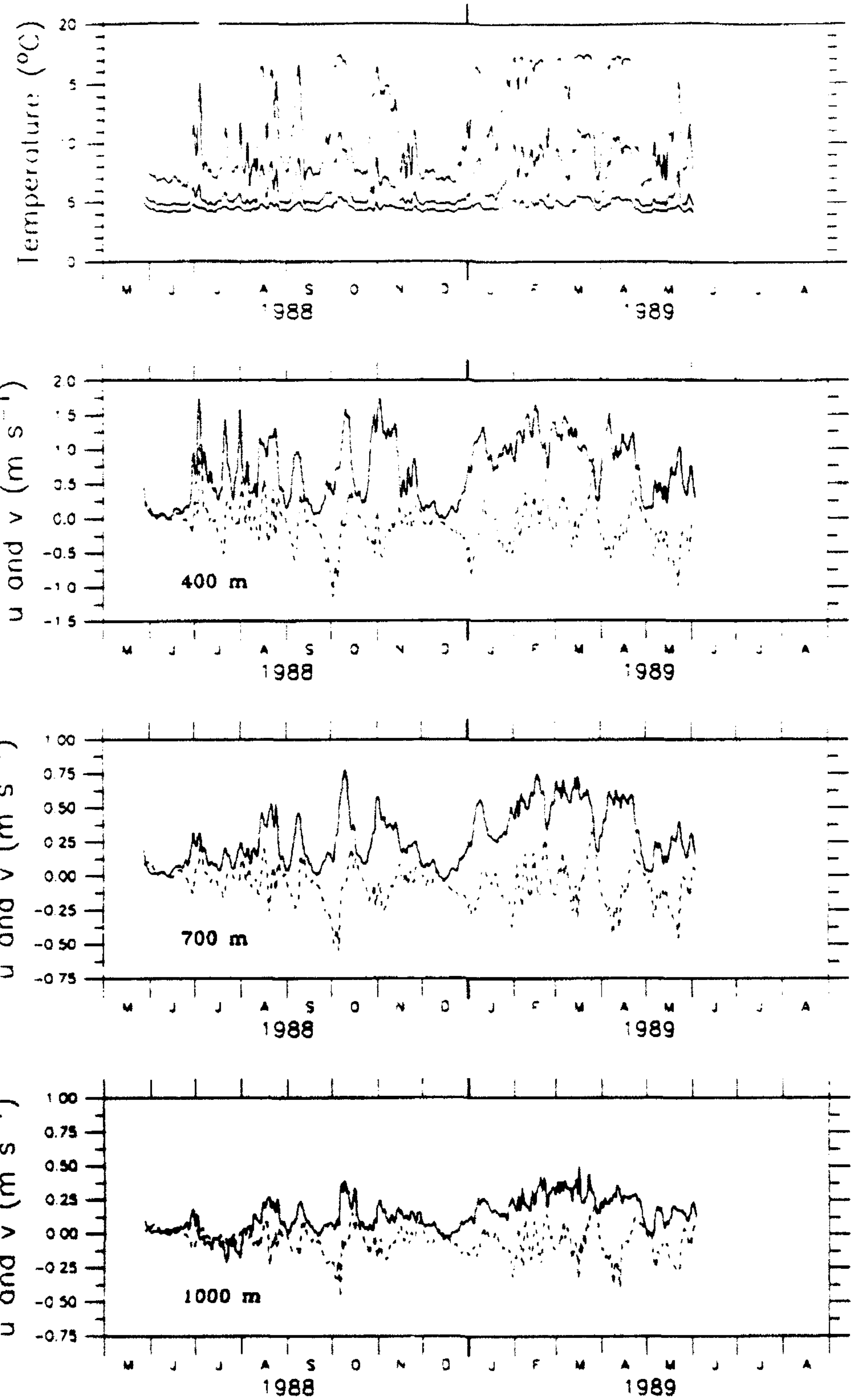
Tempertures are in units of °C. Velocity are reported in units of m s^{-1}.

Site	Data	400 m				700 m				1000 m			
	Type	Min	Max	Mean	Std	Min	Max	Mean	Std	Min	Max	Mean	Std
G2_YR1	T_{cor}	6.1923	17.5090	11.1566	3.7904	4.6593	11.2480	6.3338	1.7442	4.1247	5.5974	4.6042	0.3459
	U_{cor}	−0.0173	1.7462	0.6942	0.4493	−0.0343	0.7809	0.2811	0.2039	−0.2125	0.5030	0.1360	0.1235
	V_{cor}	−1.1491	1.0760	−0.0915	0.2900	−0.5500	0.4147	−0.0574	0.1375	−0.4708	0.2154	−0.0511	0.0976
G2_YR2	T_{cor}	4.8125	18.5380	13.2357	4.3588	4.5745	14.6640	8.5605	3.3061	4.1233	7.9212	5.3073	1.0271
	U_{cor}	−0.3116	1.8301	0.7712	0.5014	−0.3054	1.1630	0.4284	0.3011	−0.3283	0.6270	0.2096	0.1591
	V_{cor}	−0.8323	1.2777	0.1251	0.3802	−0.4849	0.8263	0.0853	0.2258	−0.2419	0.5744	0.0368	0.1316
G3_YR1	T_{cor}	8.6716	18.5160	17.1613	1.7368	5.1492	15.6810	11.8636	2.3895	4.3024	8.7245	6.2733	0.9861
	U_{cor}	0.1133	1.8738	0.7326	0.3600	0.1245	1.0788	0.5116	0.1837	0.0467	0.4884	0.2095	0.0763
	V_{cor}	−1.1208	0.8545	−0.0937	0.3301	−0.7883	0.5461	−0.0657	0.2264	−0.3452	0.1848	−0.0409	0.1036
G3_YR2	T_{cor}	5.9143	18.7860	16.5743	3.1520	4.6679	16.1540	12.2550	3.4769	4.2644	9.9745	6.8918	1.5589
	U_{cor}	−0.8225	1.5720	0.4035	0.3914	−0.5444	0.7520	0.2376	0.2085	−0.3319	0.5104	0.1190	0.1131
	V_{cor}	−1.2509	0.7700	−0.1215	0.3794	−0.6195	0.5761	−0.0525	0.2274	−0.3119	0.3352	−0.0378	0.1360
H2_YR2	T_{cor}	5.0999	17.3290	7.9408	2.5617	4.3277	10.2710	5.0763	0.8789	3.8514	5.4757	4.2773	0.2111
	U_{cor}	−0.8030	1.1766	−0.0437	0.2614	−0.1979	6.2859	−0.0281	0.0727	−0.2134	0.2070	−0.0485	0.0715
	V_{cor}	−0.7621	1.0169	−0.0038	0.1627	−0.1725	0.5447	0.0174	0.1053	−0.1366	0.2249	−0.0099	0.0440
H3_YR1	T_{cor}	5.3635	15.6520	7.1626	1.5319	4.4478	8.1114	4.8508	0.4046	3.9969	4.9398	4.2224	0.1326
	U_{cor}	−0.3809	1.1133	0.0803	0.2014	−0.1528	0.4687	0.0232	0.0855	−0.1139	0.2500	0.0085	0.0615
	V_{cor}	−0.5428	0.7217	−0.0236	0.1058	−0.1824	0.3082	−0.0246	0.0550	−0.1318	0.2087	−0.0262	0.0465
H3_YR2	T_{cor}	5.3426	18.5000	10.7966	4.8373	4.4788	13.3410	6.7209	2.7767	3.9409	6.8230	4.6430	0.7009
	U_{cor}	−0.8600	1.3756	0.3255	0.4863	−0.3866	0.8085	0.1413	0.2469	−0.3735	0.3124	0.0471	0.1245
	V_{cor}	−1.1025	1.2135	0.0282	0.2408	−0.4625	0.6083	0.0444	0.1523	−0.2603	0.2577	−0.0045	0.0740
H4_YR1	T_{cor}	5.7710	18.0220	12.5852	4.0202	4.4950	14.2870	7.4767	2.6161	4.0753	7.6128	4.8669	0.6989
	U_{cor}	−0.1485	1.6689	0.7806	0.4948	−0.2264	0.4105	0.1408	0.1434	−0.2063	0.9430	0.3415	0.2794
	V_{cor}	−1.0076	1.1265	−0.0085	0.3683	−0.5107	0.6696	−0.0388	0.1970	−0.3372	0.3876	−0.0491	0.1325
H4_YR2	T_{cor}	5.6784	18.8380	13.7716	5.1400	4.3193	15.1720	9.1872	3.9771	3.8903	8.5451	5.5624	1.5061
	U_{cor}	−0.7213	1.7687	0.4054	0.4946	−0.3170	0.8982	0.2144	0.2733	−0.2697	0.4803	0.0937	0.1577
	V_{cor}	−1.2958	0.7185	−0.0680	0.2661	−0.6494	0.4827	−0.0389	0.1631	−0.3511	0.2717	−0.0508	0.1116
H5_YR1	T_{cor}	6.3924	18.7090	16.1667	3.8544	4.6485	15.7400	12.0380	3.6077	4.1076	9.1186	6.7266	1.1118
	U_{cor}	−0.0218	1.4843	0.4222	0.2578	−0.1051	0.8399	0.2965	0.1673	−0.1186	0.1748	0.1660	0.1025
	V_{cor}	−1.5340	0.6293	−0.0972	0.2554	−0.8798	0.3670	−0.0806	0.1776	−0.4071	0.1891	−0.0682	0.1337

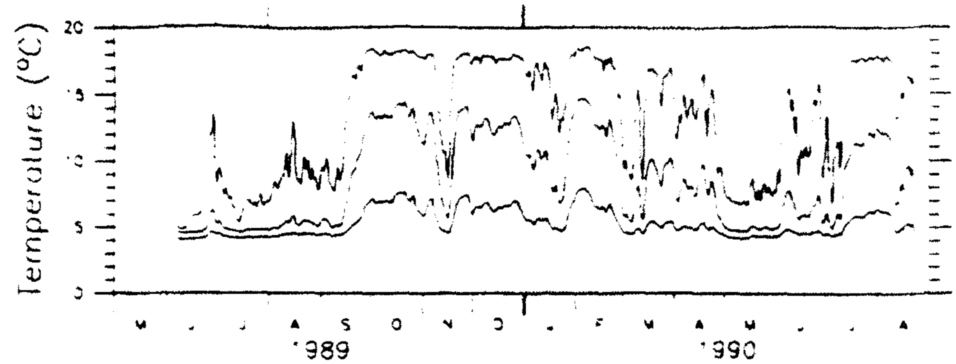
Table 11. (continued)

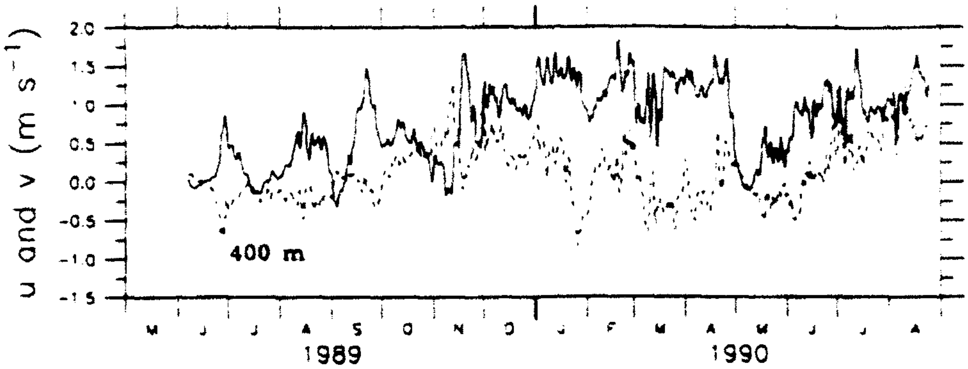
Site	Data Type	400 m Min	400 m Max	400 m Mean	400 m Std	700 m Min	700 m Max	700 m Mean	700 m Std	1000 m Min	1000 m Max	1000 m Mean	1000 m Std
H6_YR2	T_{cor}	6.1320	18.8880	17.0001	2.6815	4.6351	17.3680	13.2834	3.4610	4.1564	12.0930	7.6403	1.7693
	U_{cor}	−1.1379	1.5808	0.1366	0.4268	−0.5594	0.8825	0.1141	0.2673	−0.4746	0.5098	0.0839	0.1584
	V_{cor}	−1.9143	0.9843	−0.2233	0.4380	−1.0763	0.5270	−0.1343	0.2499	−0.6174	0.2406	−0.0752	0.1374
I1_YR2	T_{cor}	5.2503	17.5910	7.9121	2.7046	4.2076	10.8570	5.1156	1.0246	3.9104	5.3392	4.2743	0.2280
	U_{cor}	−0.6952	1.2058	−0.0314	0.2352	−0.2698	0.5312	−0.0391	0.0969	−0.2057	0.1781	−0.0458	0.0627
	V_{cor}	−0.6262	0.8523	−0.0310	0.1702	−0.2985	0.3863	−0.0289	0.0656	−0.1517	0.1641	−0.0271	0.0412
I2_YR1	T_{cor}	5.3115	15.0390	6.7878	0.9672	4.3820	7.5242	4.7724	0.2580	3.8250	4.8568	4.1837	0.1057
	U_{cor}	−0.2029	0.8147	0.0216	0.1379	−0.2060	0.3327	−0.0074	0.0797	−0.2124	0.2024	−0.0175	0.0724
	V_{cor}	−0.3669	0.5258	−0.0135	0.0807	−0.1312	0.1834	−0.0181	0.0523	−0.1368	0.1111	−0.0180	0.0469
I2_YR2	T_{cor}	5.3072	18.4950	10.5605	4.7043	4.4233	13.3950	6.5176	2.6467	4.0199	7.0098	4.6227	0.6966
	U_{cor}	−1.0992	1.5707	0.2098	0.5172	−0.5433	0.7035	0.0881	0.2577	−0.3763	0.3412	0.0142	0.1380
	V_{cor}	−0.5646	0.9331	0.0238	0.2779	−0.2963	0.3309	0.0000	0.1208	−0.1737	0.1799	−0.0080	0.0785
I3_YR1	T_{cor}	5.5390	17.9670	10.5394	3.9682	4.4750	12.1640	6.1340	1.8758	3.9028	6.0990	4.5296	0.3947
	U_{cor}	−0.2661	1.6100	0.6117	0.5130	−0.3834	0.8115	0.2532	0.2596	−0.4713	0.5366	0.1287	0.1666
	V_{cor}	−0.7776	1.0290	0.0316	0.2989	−0.3995	0.4789	0.0093	0.1675	−0.3994	0.3336	−0.0029	0.1280
I3_YR2	T_{cor}	5.7862	18.5560	13.6118	4.9259	4.5688	14.6490	9.3390	3.9716	4.1128	8.0618	5.6310	1.3675
	U_{cor}	−1.2230	2.0587	0.3900	0.6264	−0.4881	1.3945	0.2043	0.3507	−0.4410	0.5746	0.0753	0.1564
	V_{cor}	−1.5187	0.9859	−0.1235	0.2861	−0.5047	0.4942	−0.0738	0.1613	−0.2579	0.1847	−0.0532	0.0882
I4_YR1	T_{cor}	5.3088	18.5110	15.7308	4.0383	4.4762	15.3450	11.2777	3.5903	4.1240	8.6996	6.3102	1.4620
	U_{cor}	−0.2156	1.4731	0.5201	0.3755	−0.2965	1.0242	0.3749	0.2731	−0.2076	0.5112	0.1658	0.1289
	V_{cor}	−1.0326	1.2270	0.0340	0.3184	−0.4838	0.7412	0.0591	0.2149	−0.2270	0.3716	0.0550	0.1131
I4_YR2	T_{cor}	5.6137	18.9410	15.7150	3.9695	4.7543	16.5690	11.2615	3.8646	4.1647	9.8712	6.5235	1.7142
	U_{cor}	−1.0033	1.6109	0.3352	0.5142	−0.6719	0.9921	0.2339	0.3406	−0.2857	0.4475	0.1113	0.1521
	V_{cor}	−1.1912	1.4702	−0.0483	0.3386	−0.7862	0.9003	−0.0522	0.2330	−0.3424	0.5781	−0.0222	0.1322
I5_YR2	T_{cor}	5.8171	18.8610	16.7243	3.2186	4.6837	18.0330	12.8192	3.3836	4.1933	12.1160	7.4533	1.7353
	U_{cor}	−1.1128	1.6603	0.2013	0.4179	−0.7221	0.8955	0.1433	0.2773	−0.4518	0.5372	0.0891	0.1604
	V_{cor}	−1.4294	1.7882	0.1193	0.3487	−0.6824	0.8192	0.0744	0.2218	−0.3470	0.5185	0.0558	0.1397
M13_YR2	T_{cor}	6.0476	18.8780	15.7366	3.9988	4.6583	16.3050	11.1235	3.8600	4.1022	9.7169	6.4188	1.6615
	U_{cor}	−0.9429	1.5084	0.3751	0.5247	−0.5883	1.1459	0.2316	0.3259	−0.2492	0.5291	0.0985	0.1505
	V_{cor}	−1.1452	0.9195	−0.1494	0.3281	−0.6526	0.6175	−0.1043	0.2183	−0.3659	0.2445	−0.0685	0.1119

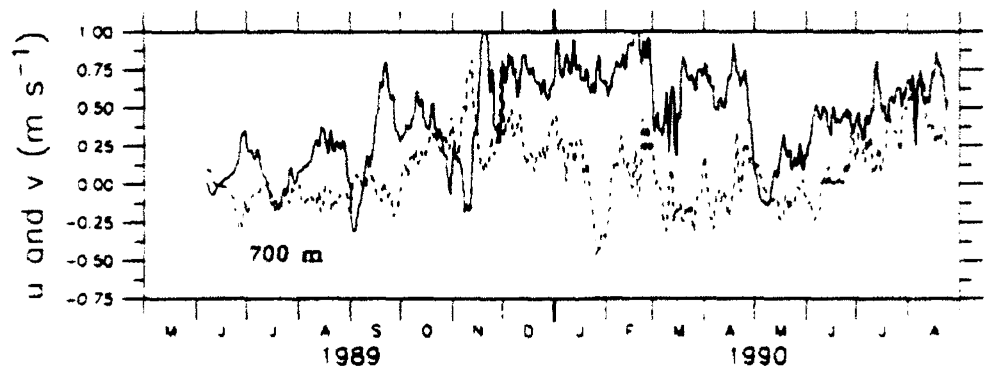
Site G2 Year 1

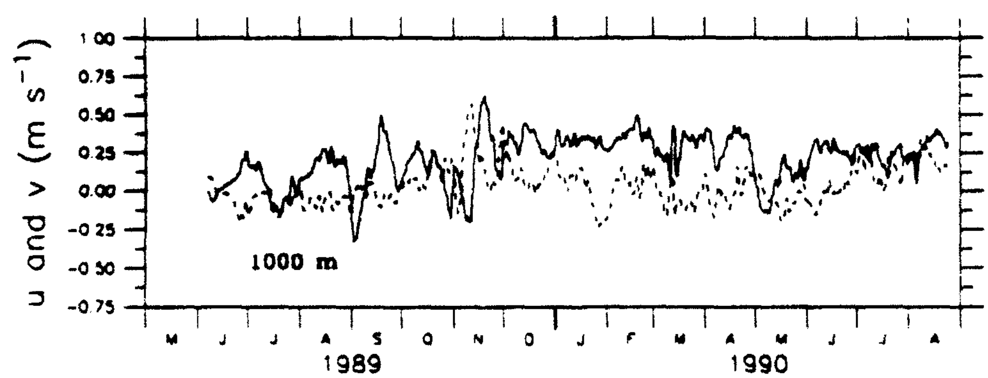


Site G2 Year 2

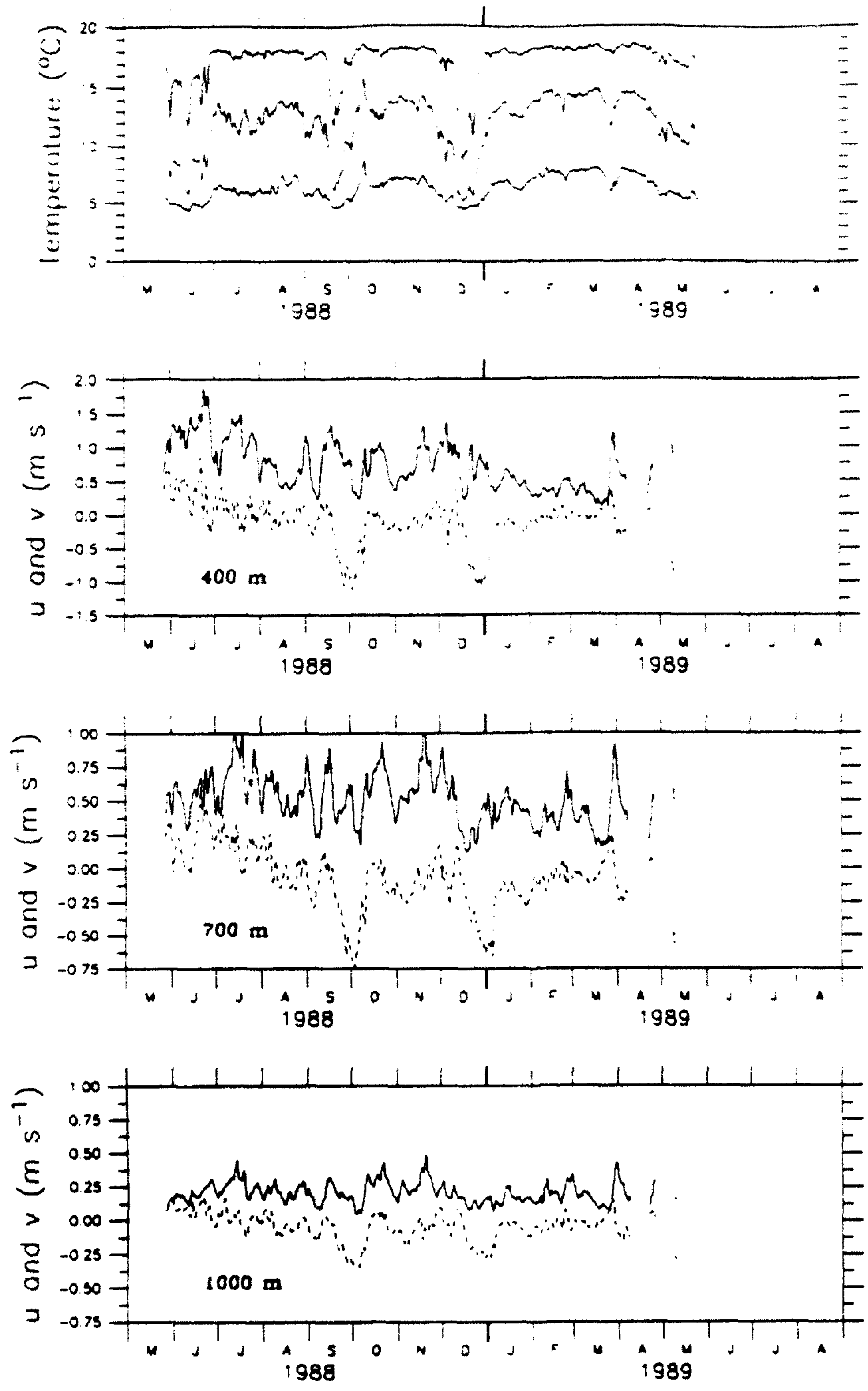




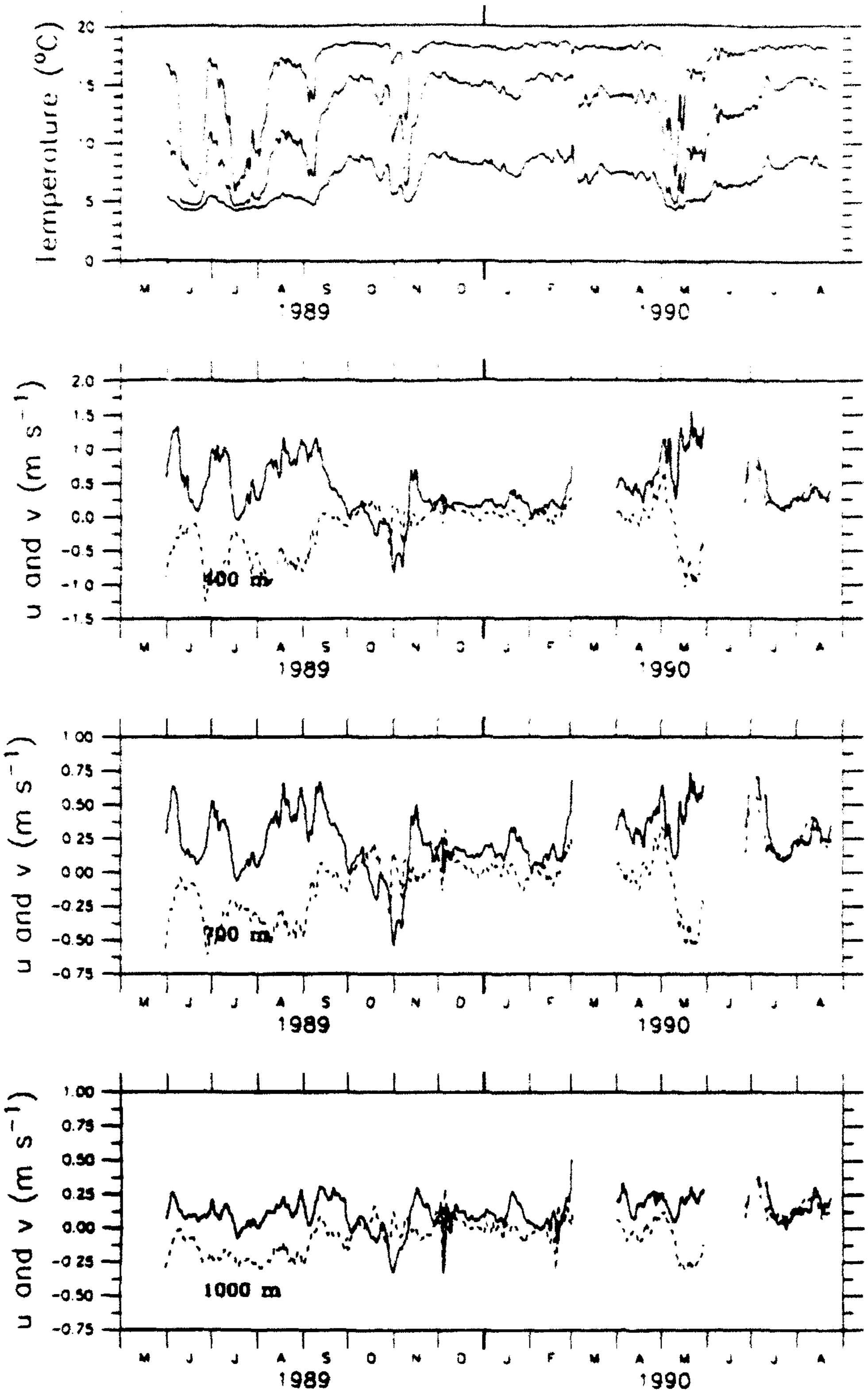




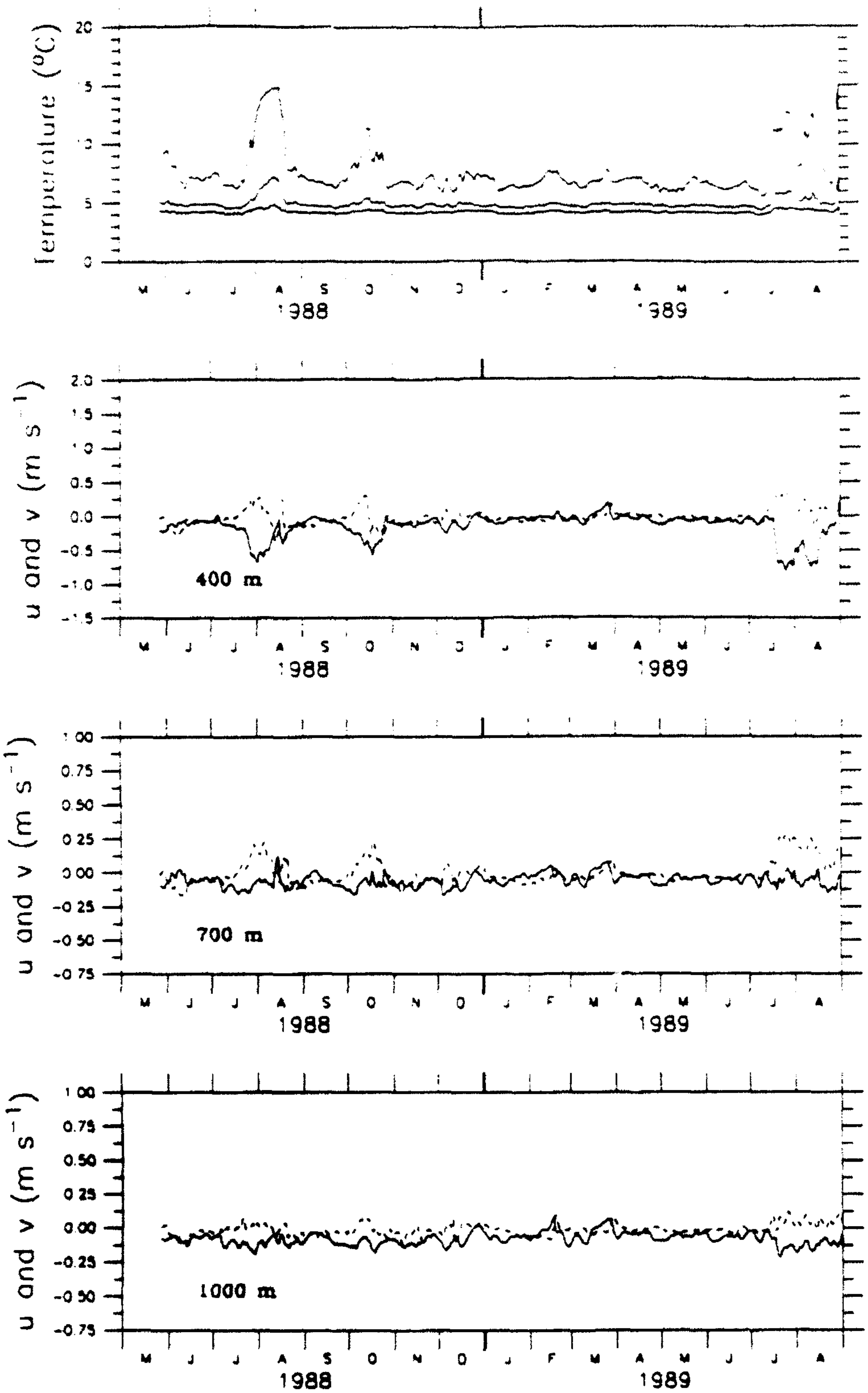
Site G3 Year 1



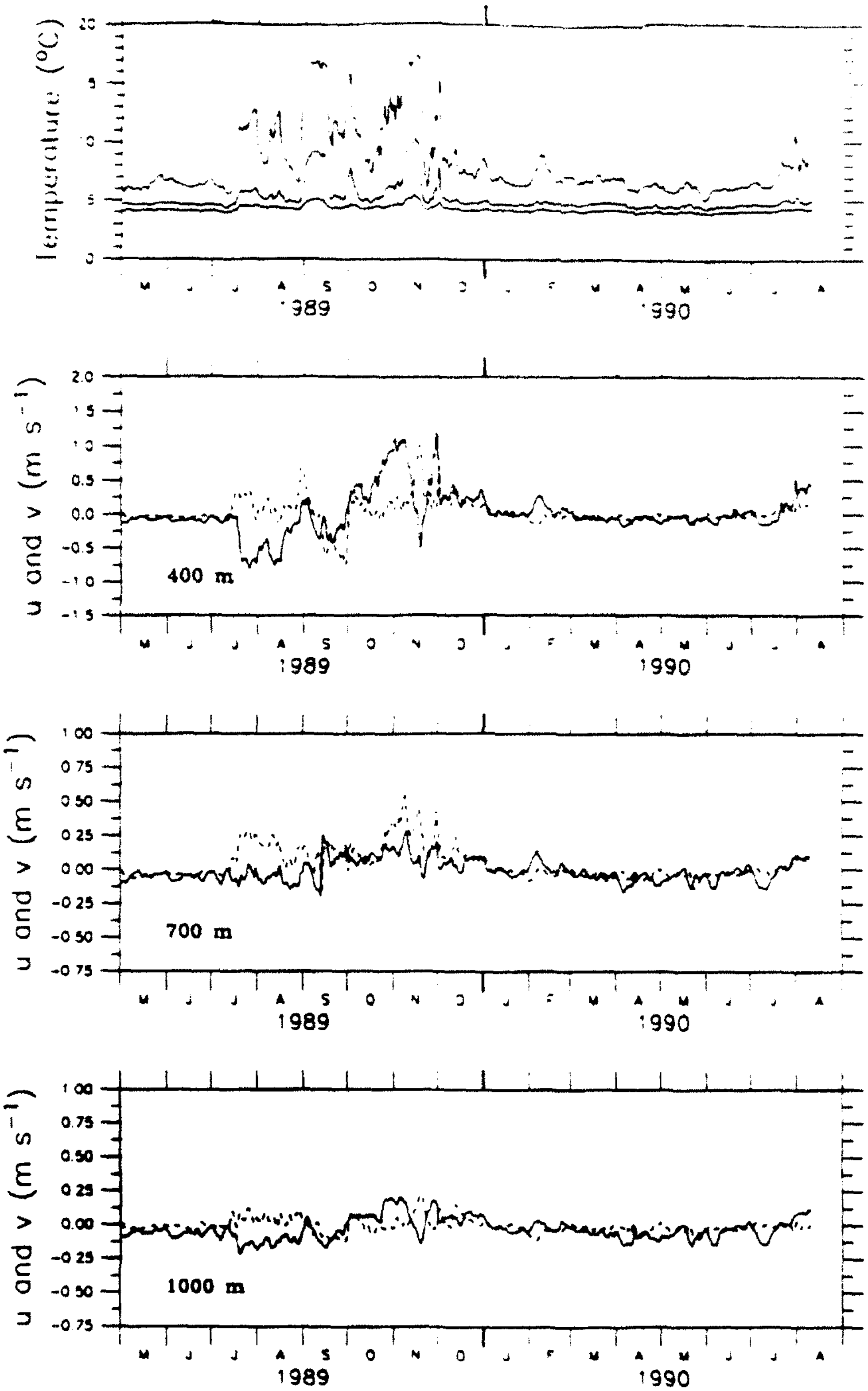
Site G3 Year 2



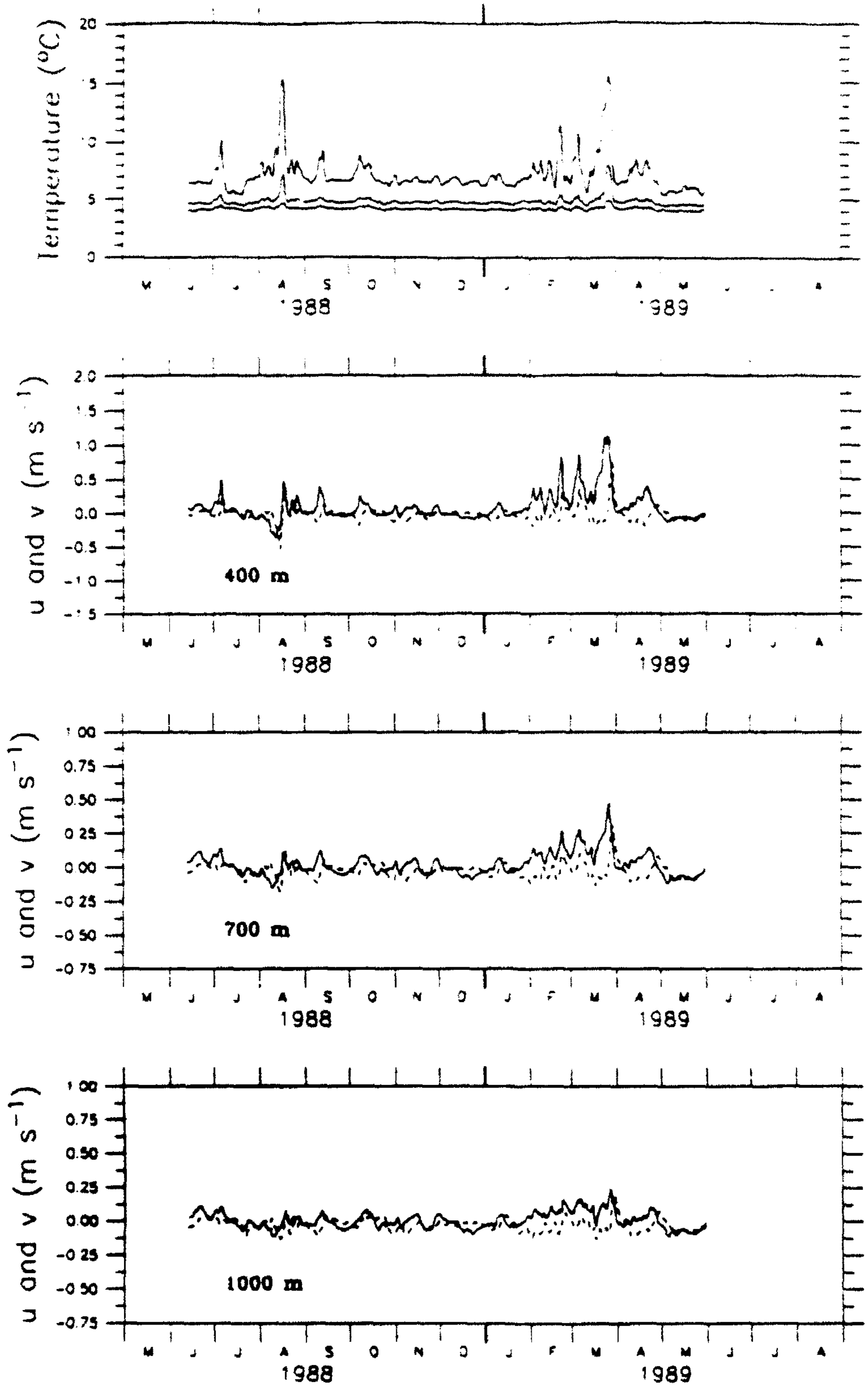
Site H2 Year 1



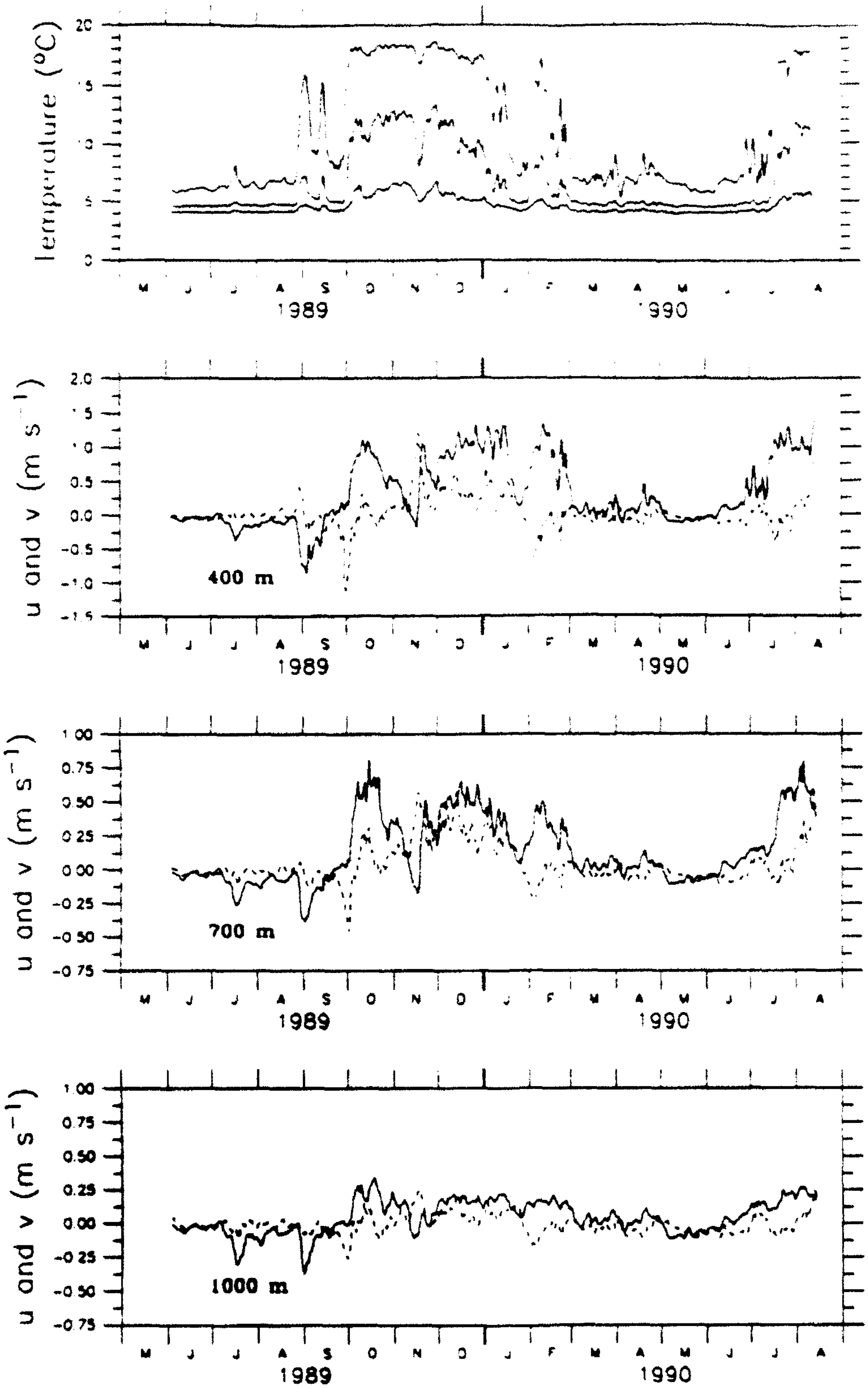
Site H2 Year 2



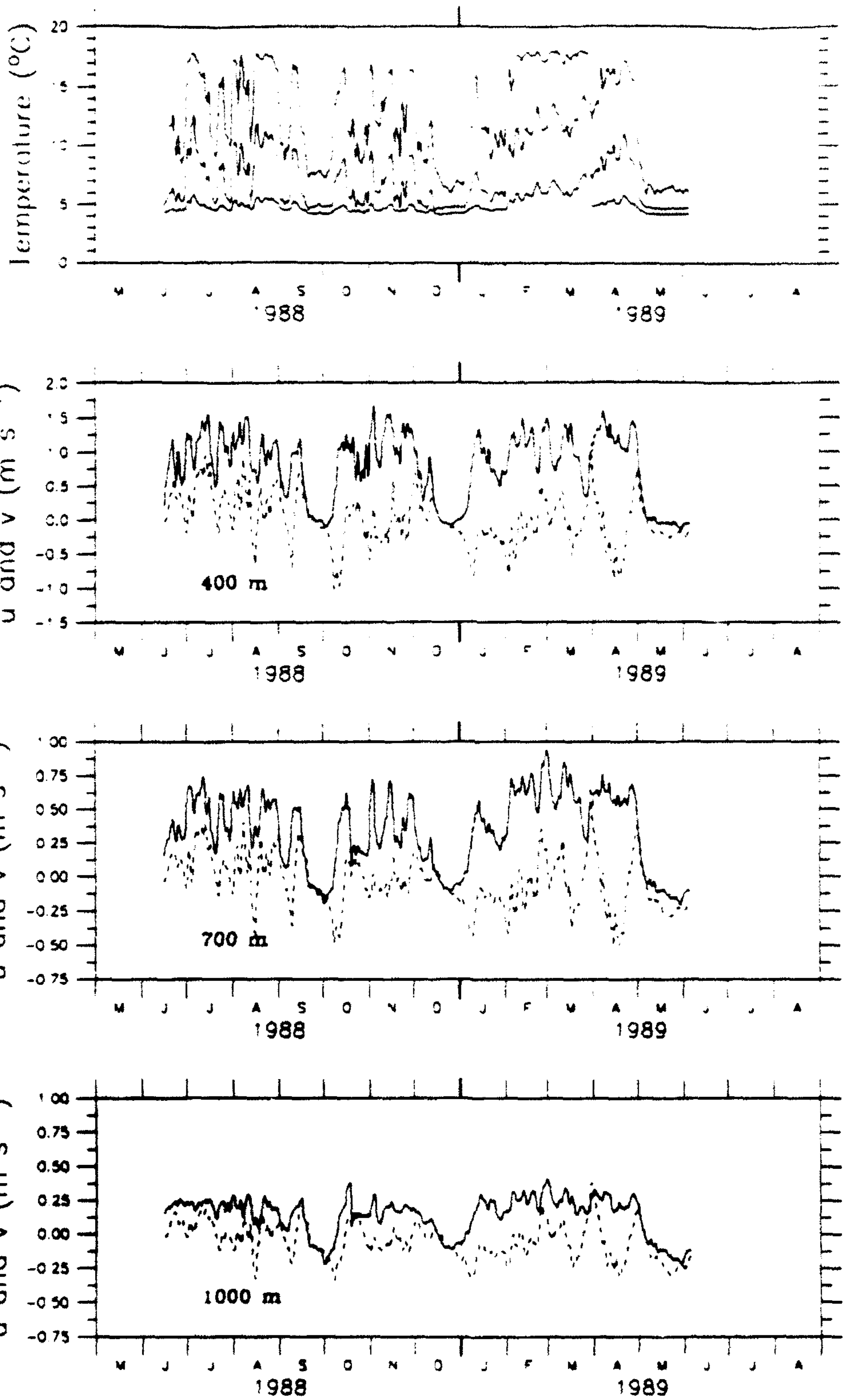
Site H3 Year 1



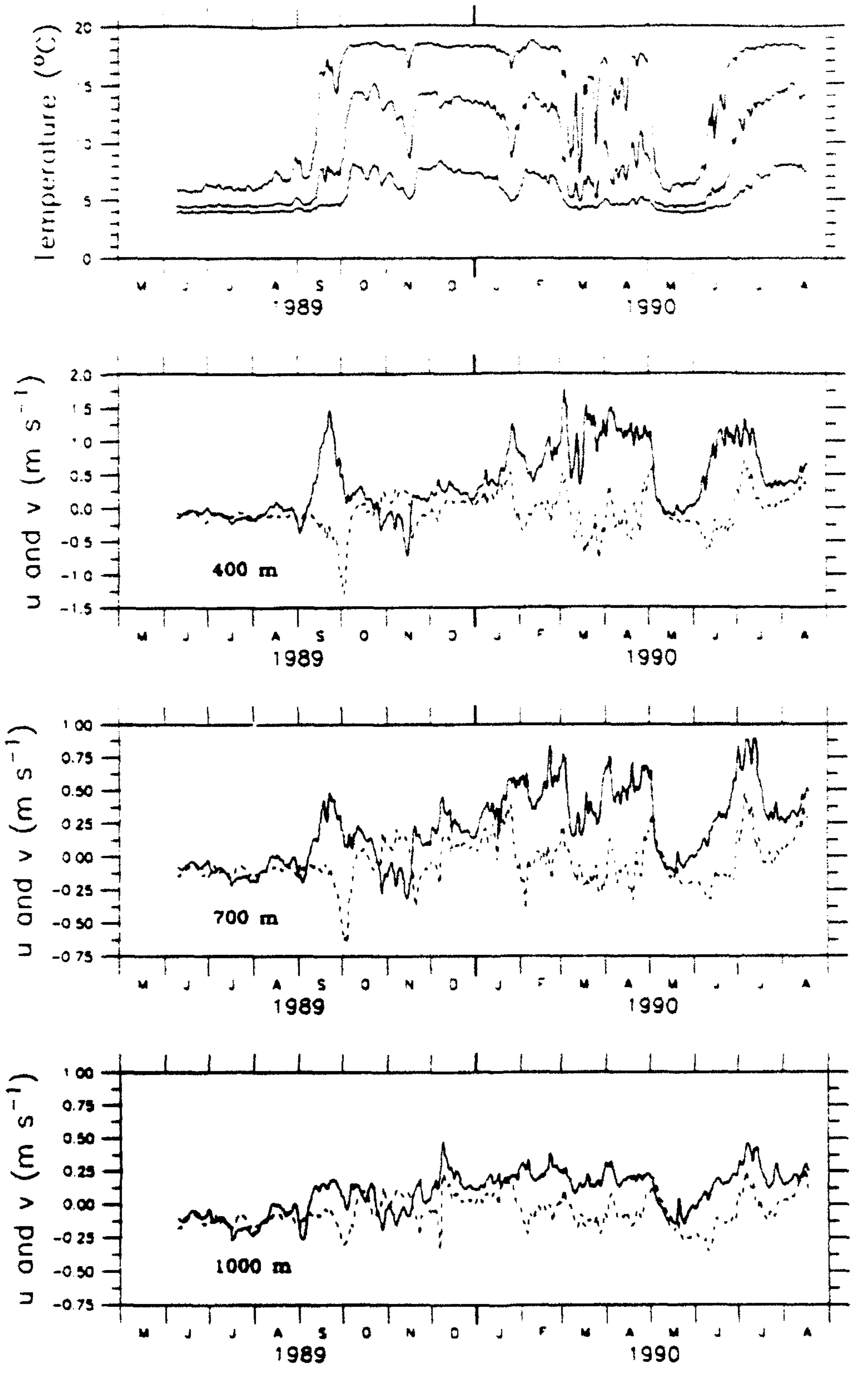
Site H3 Year 2



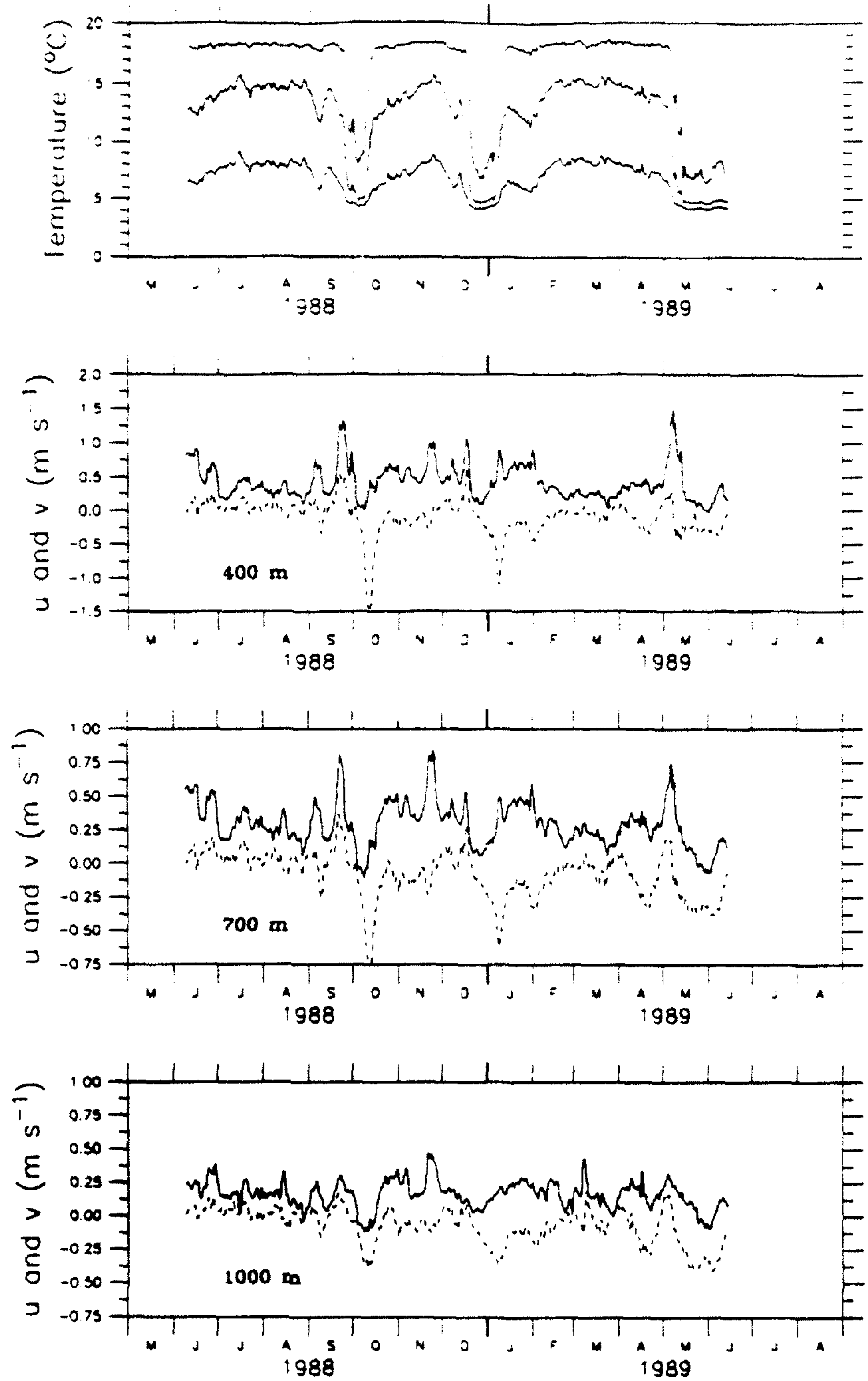
Site H4 Year 1



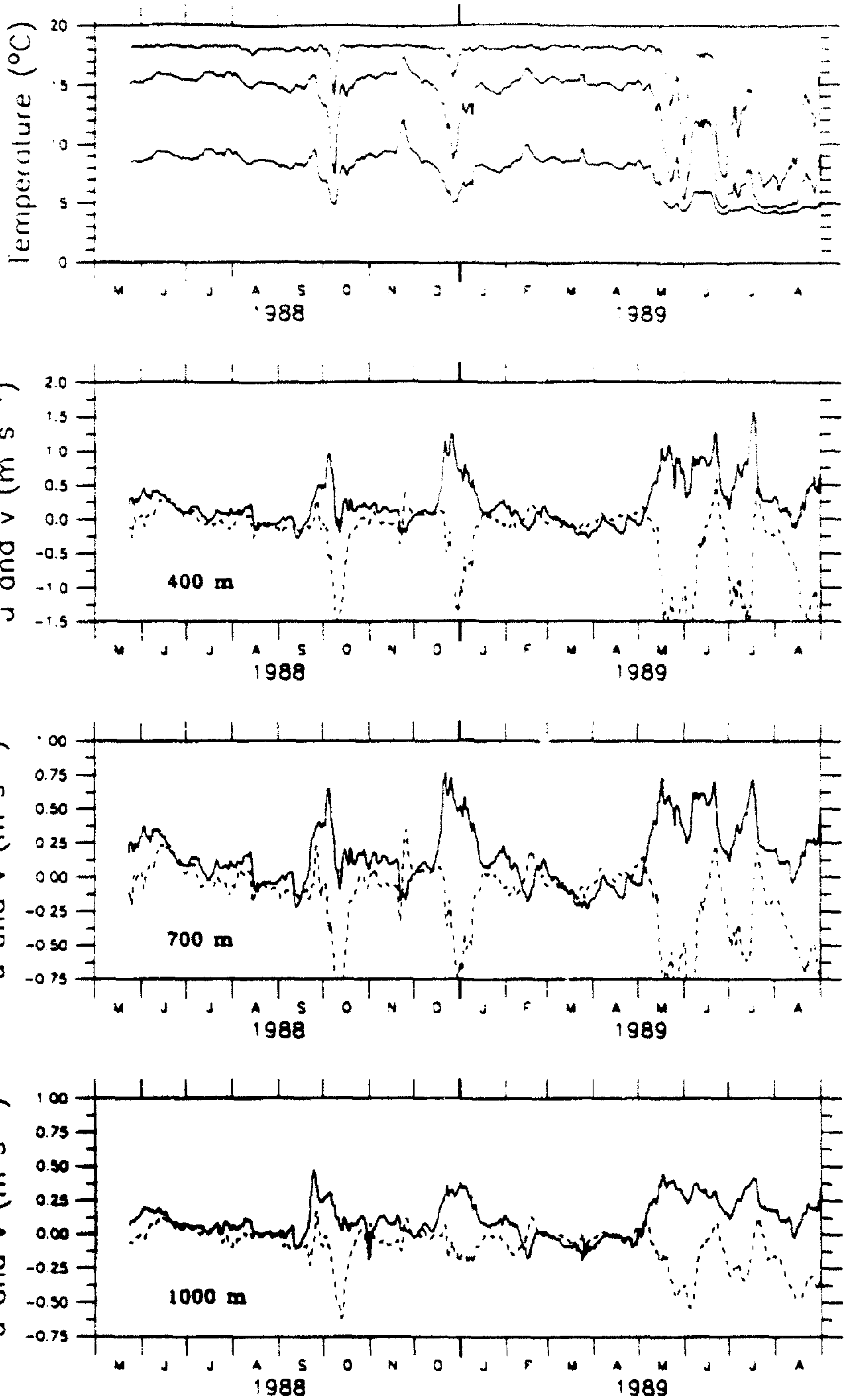
Site H4 Year 2



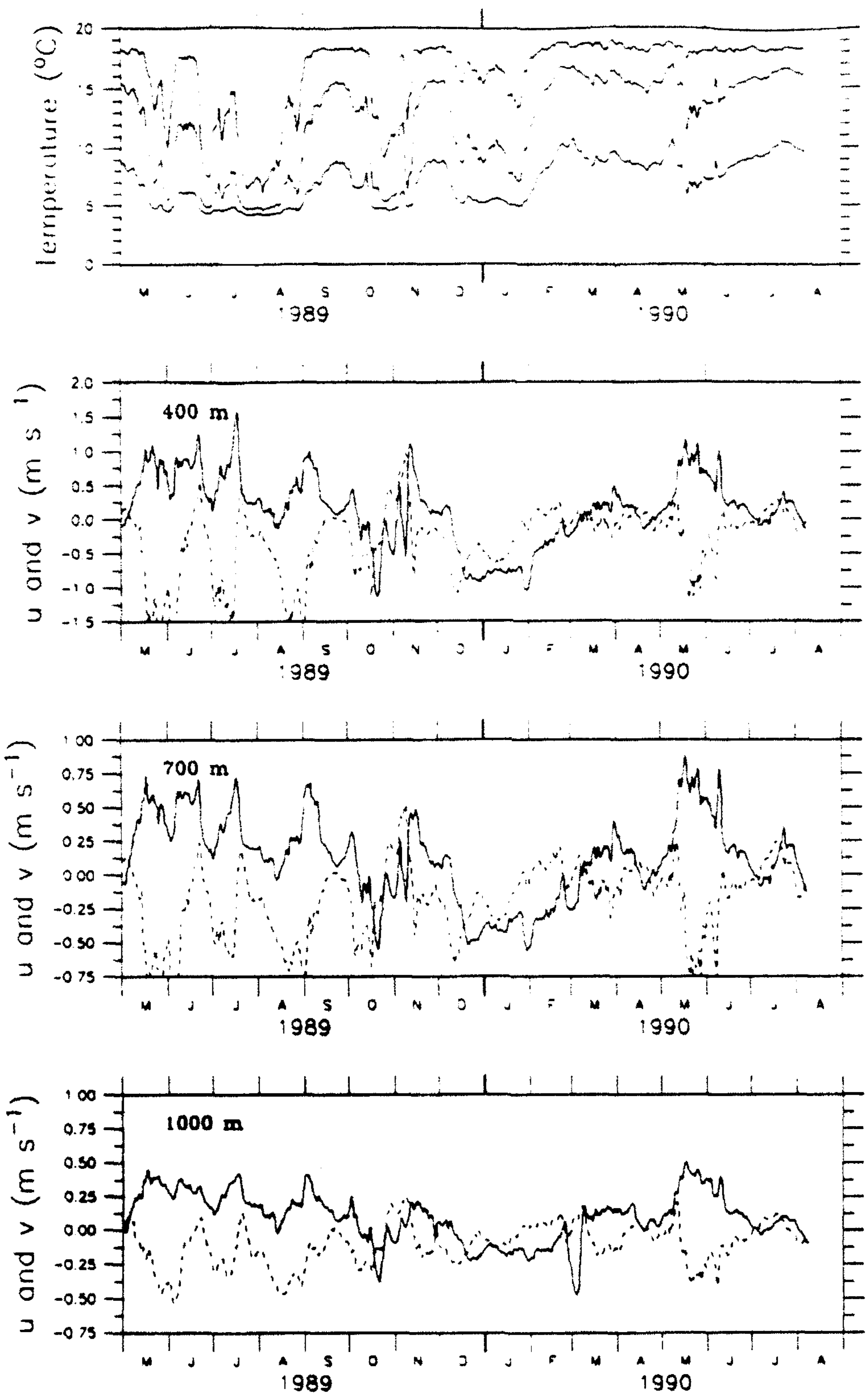
Site H5 Year 1



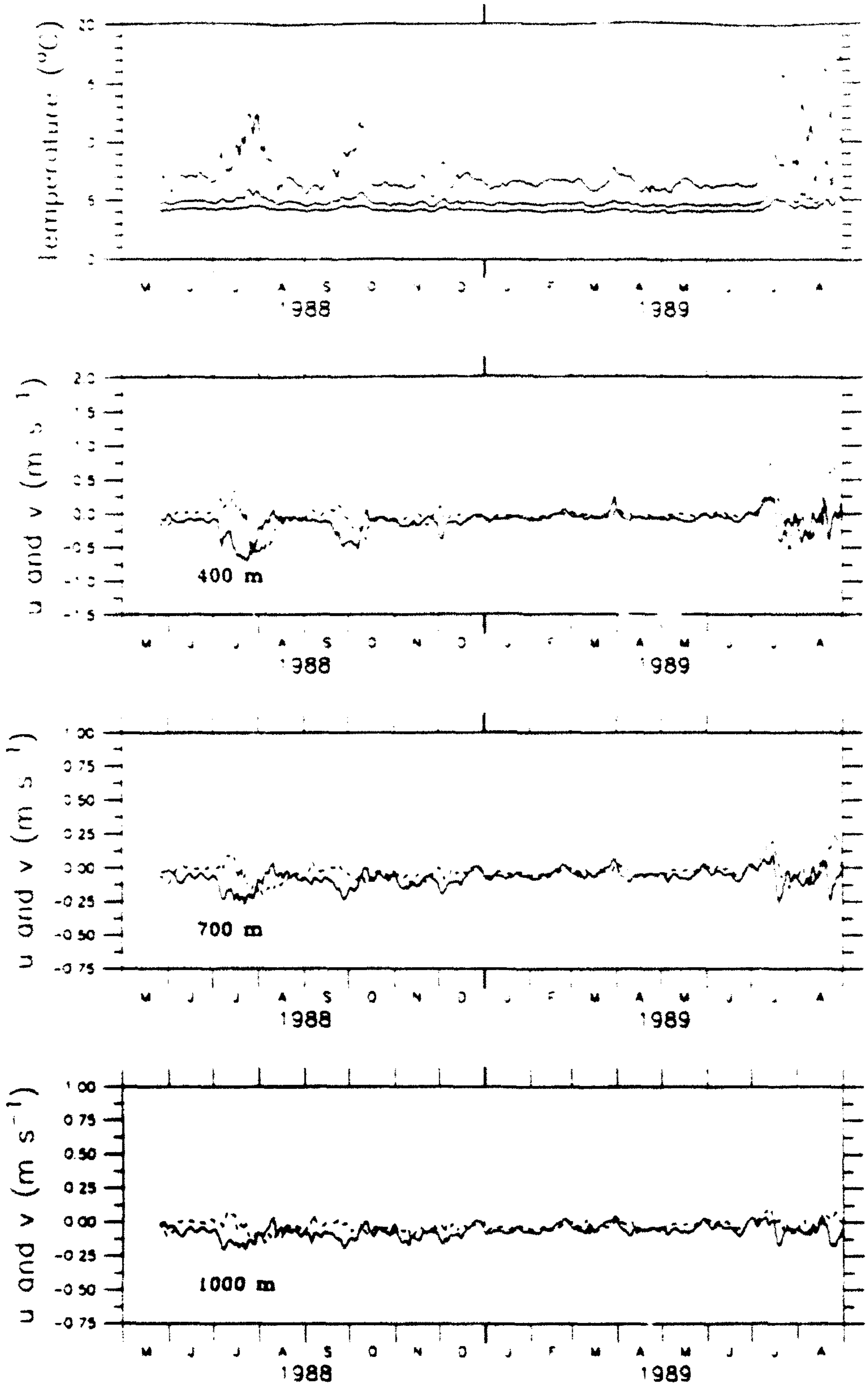
Site H6 Year 1



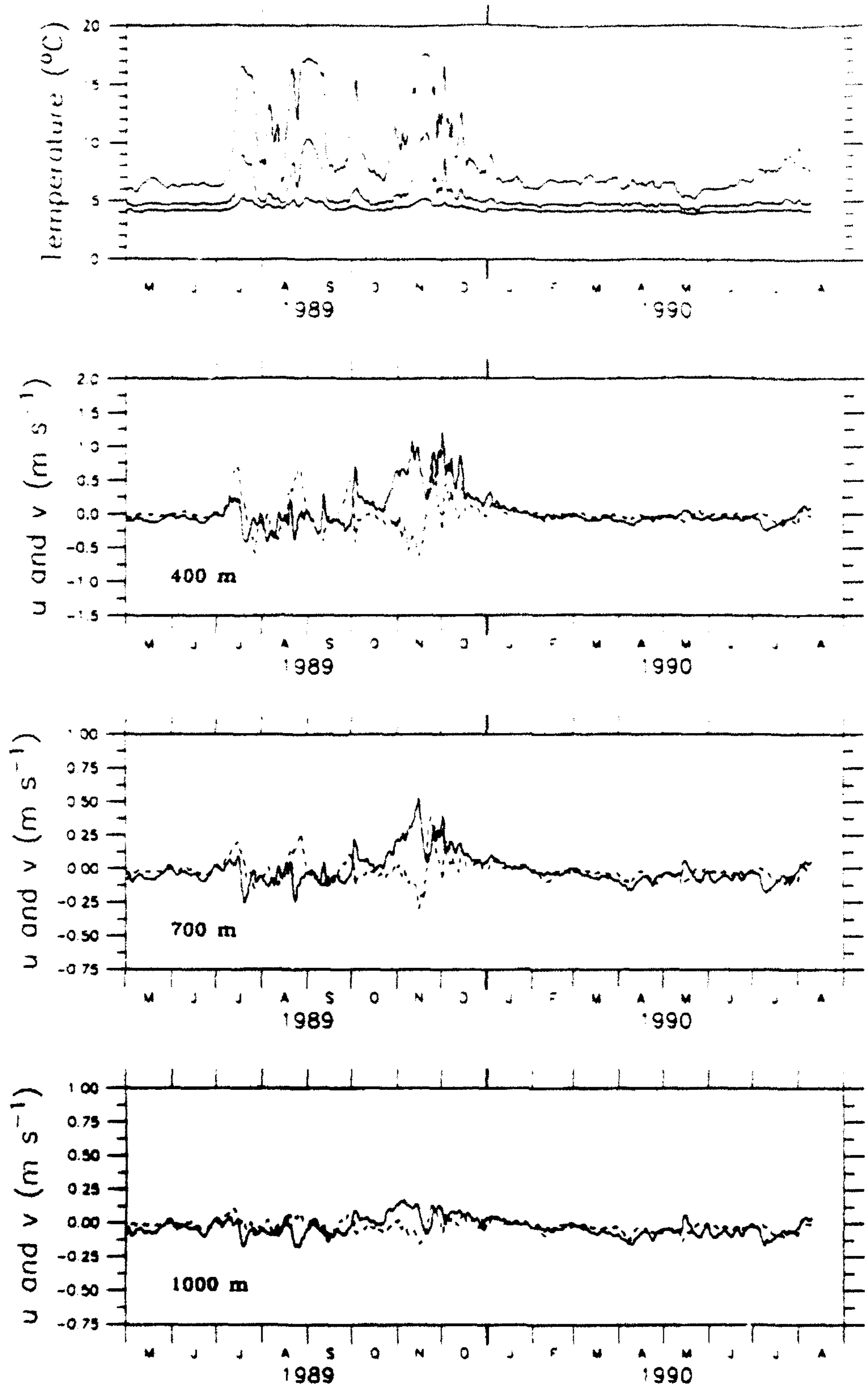
Site H6 Year 2



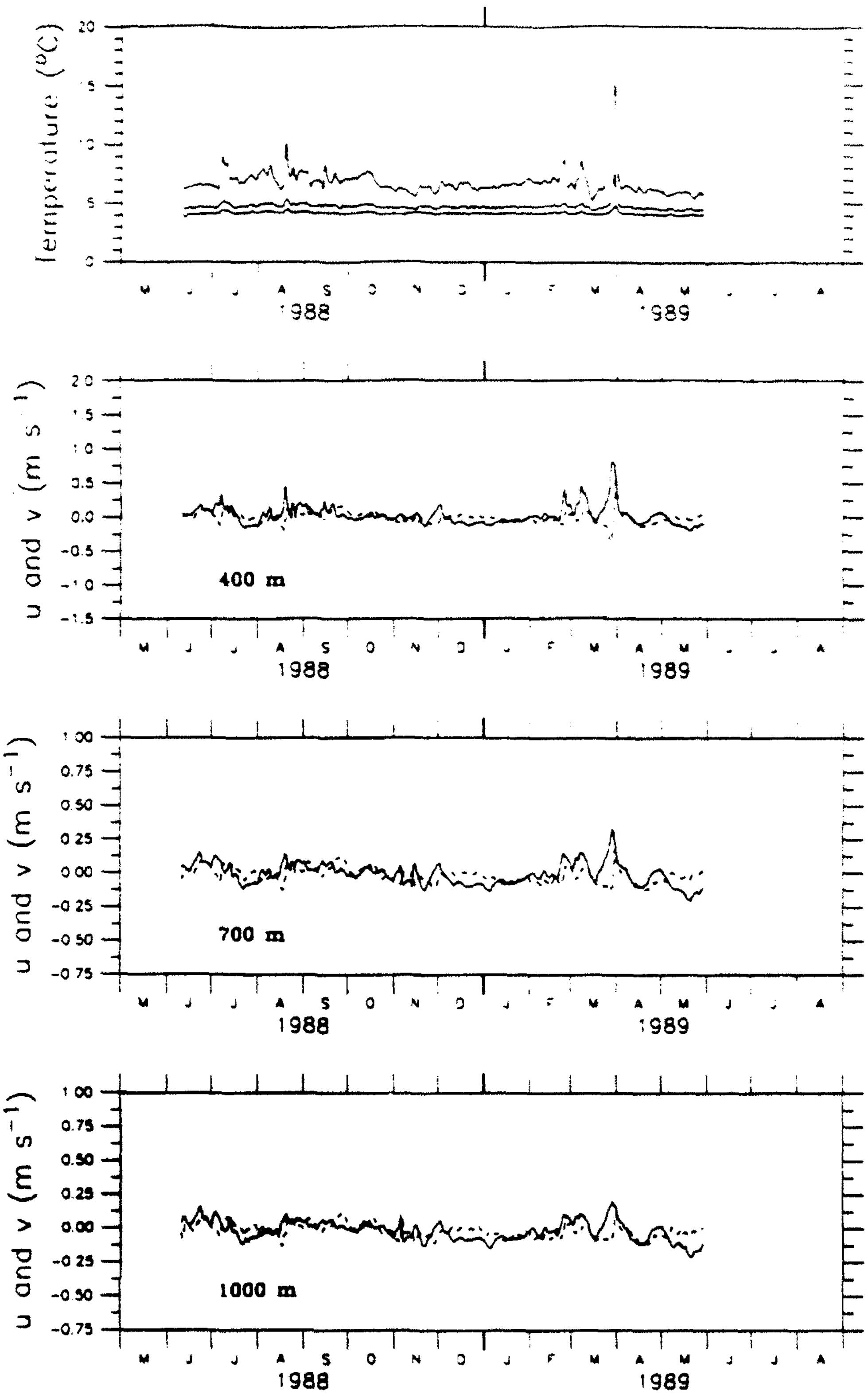
Site II Year 1



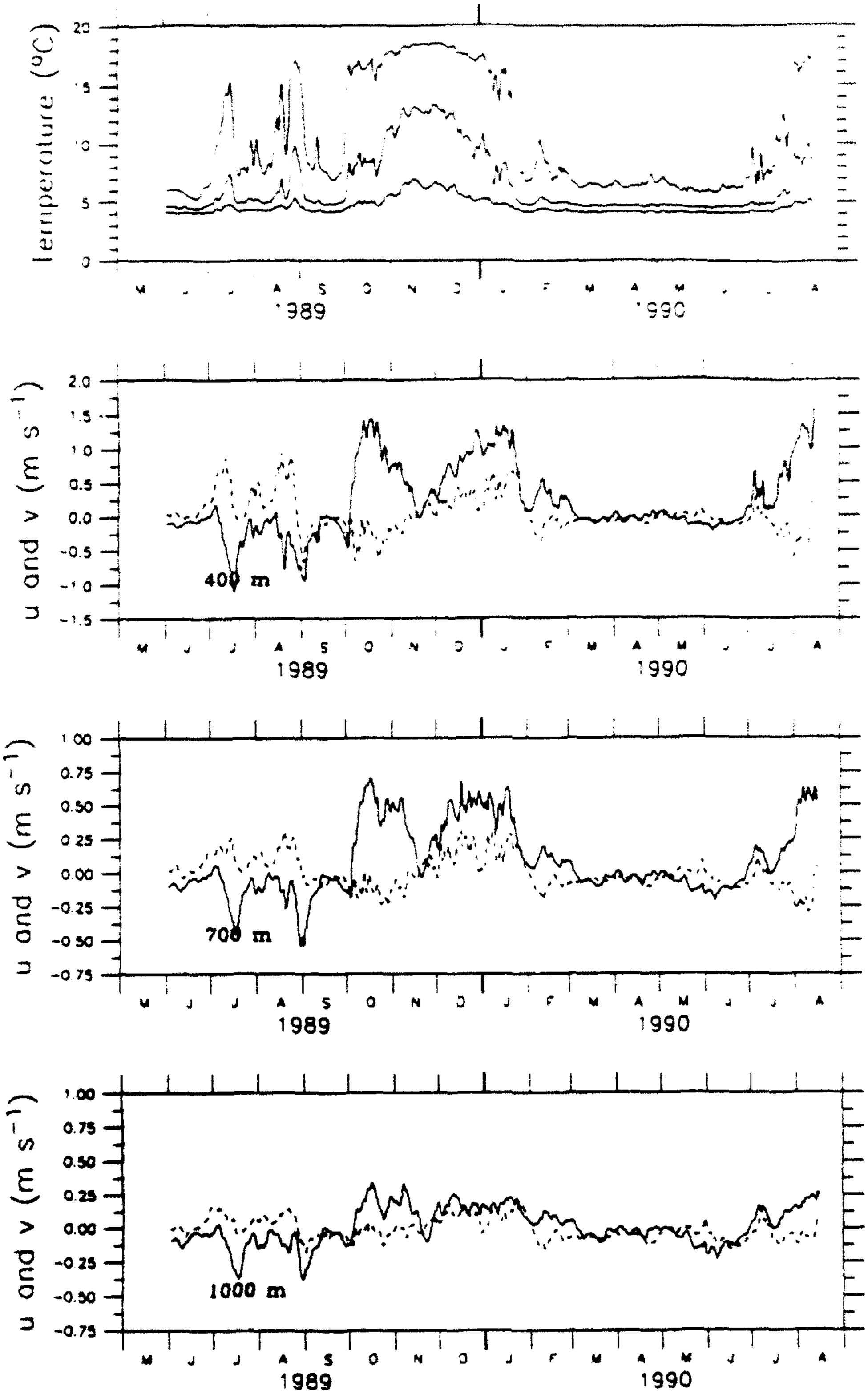
Site II Year 2



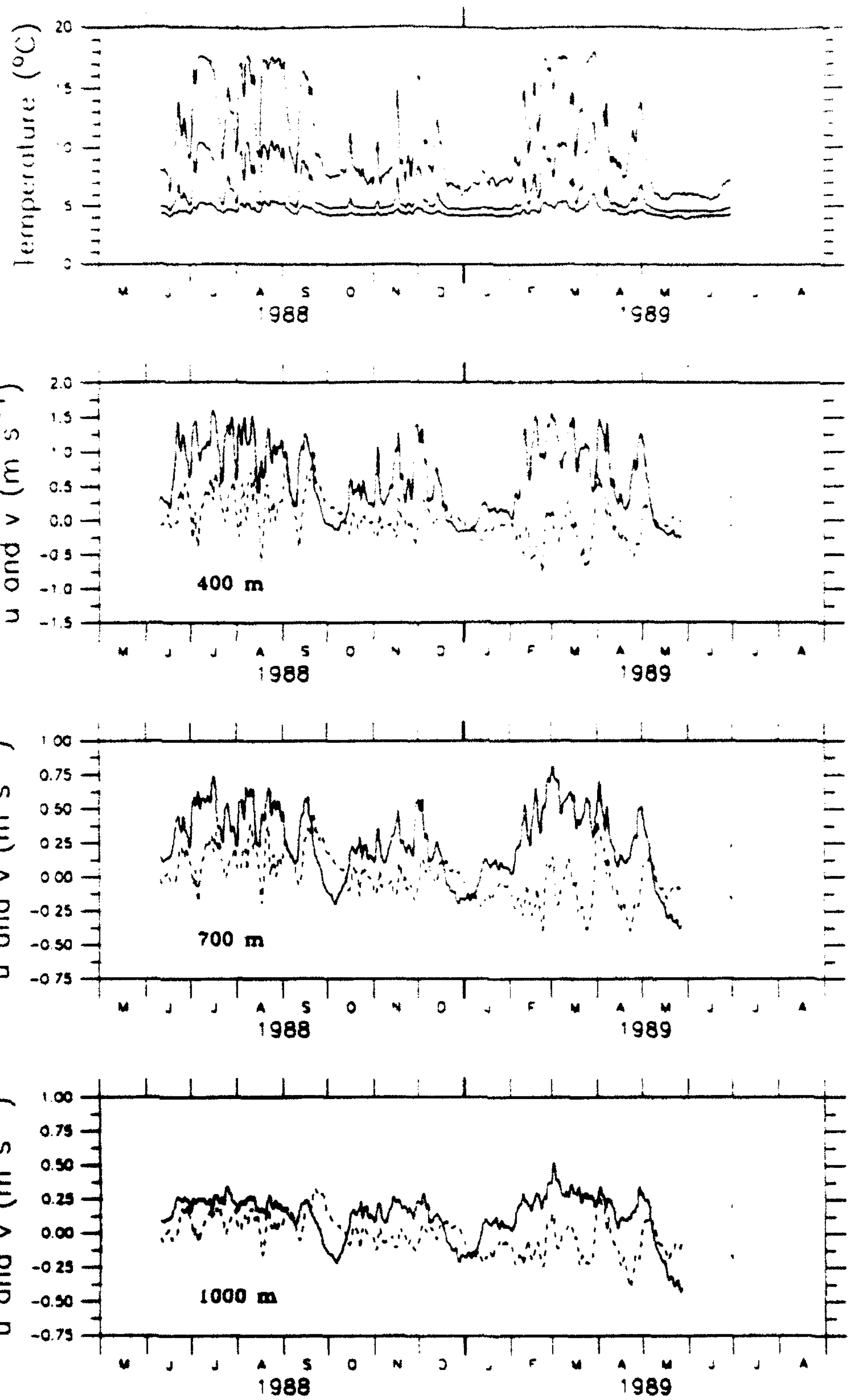
Site I2 Year 1



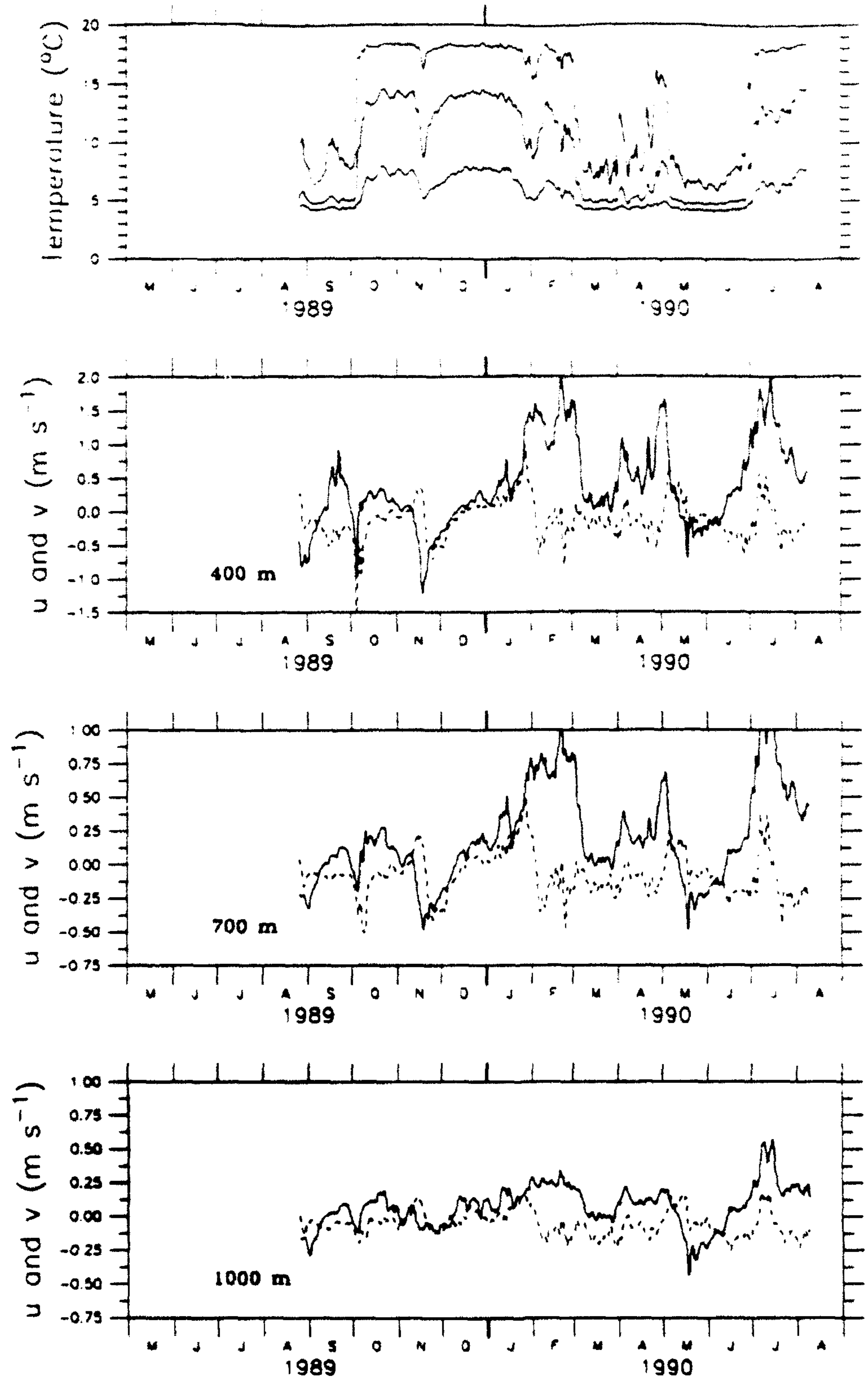
Site I2 Year 2



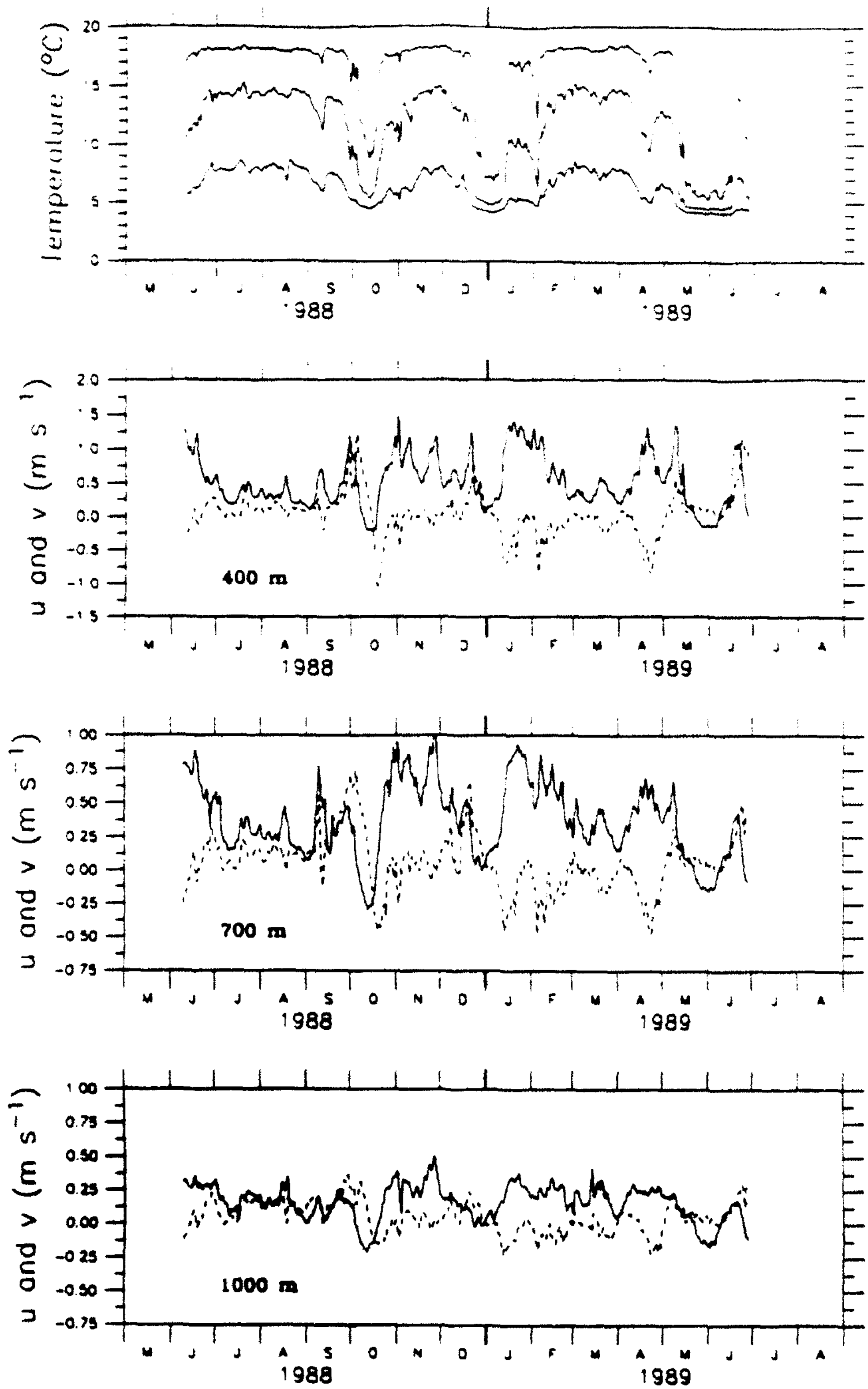
Site 13 Year 1



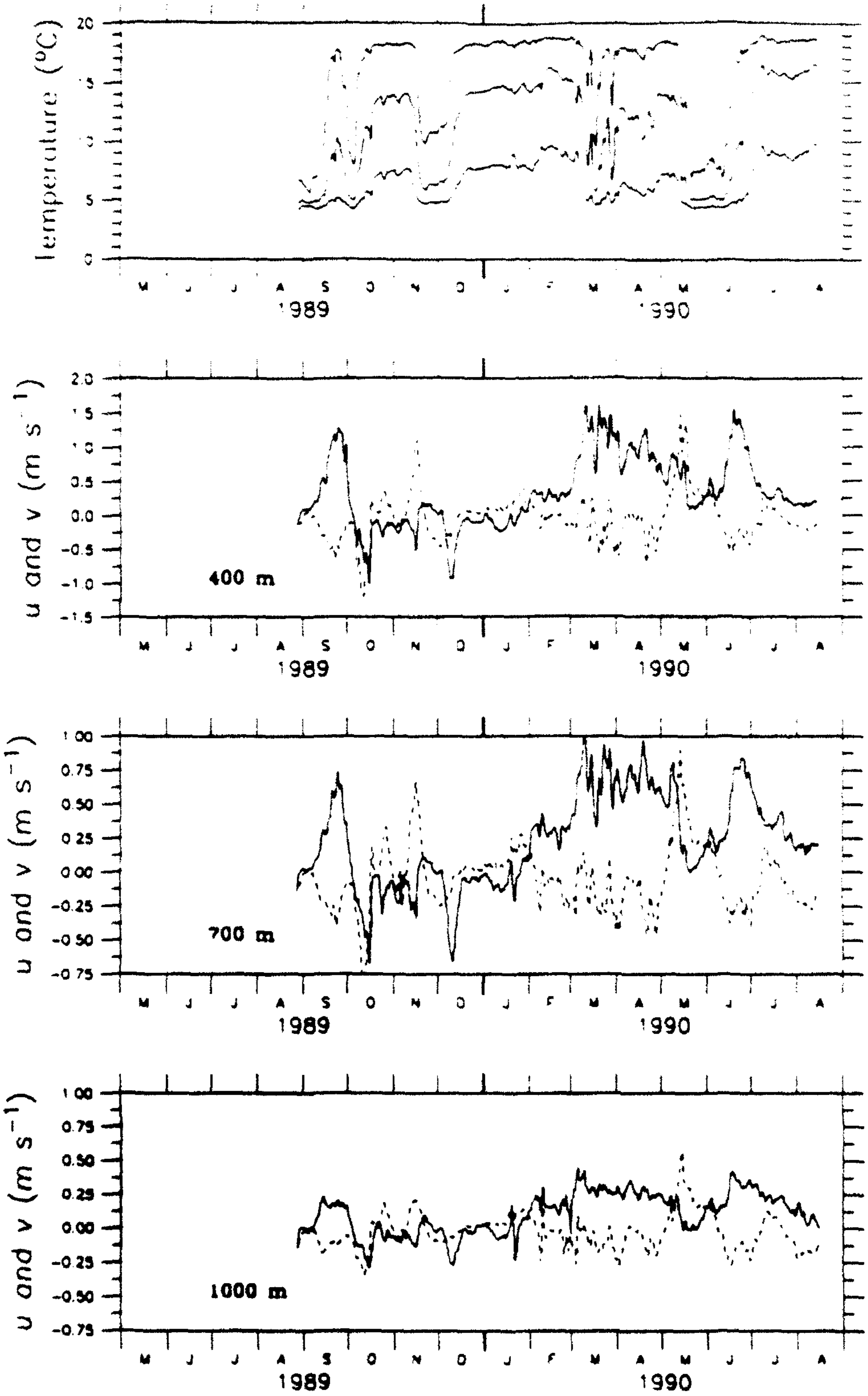
Site I3 Year 2



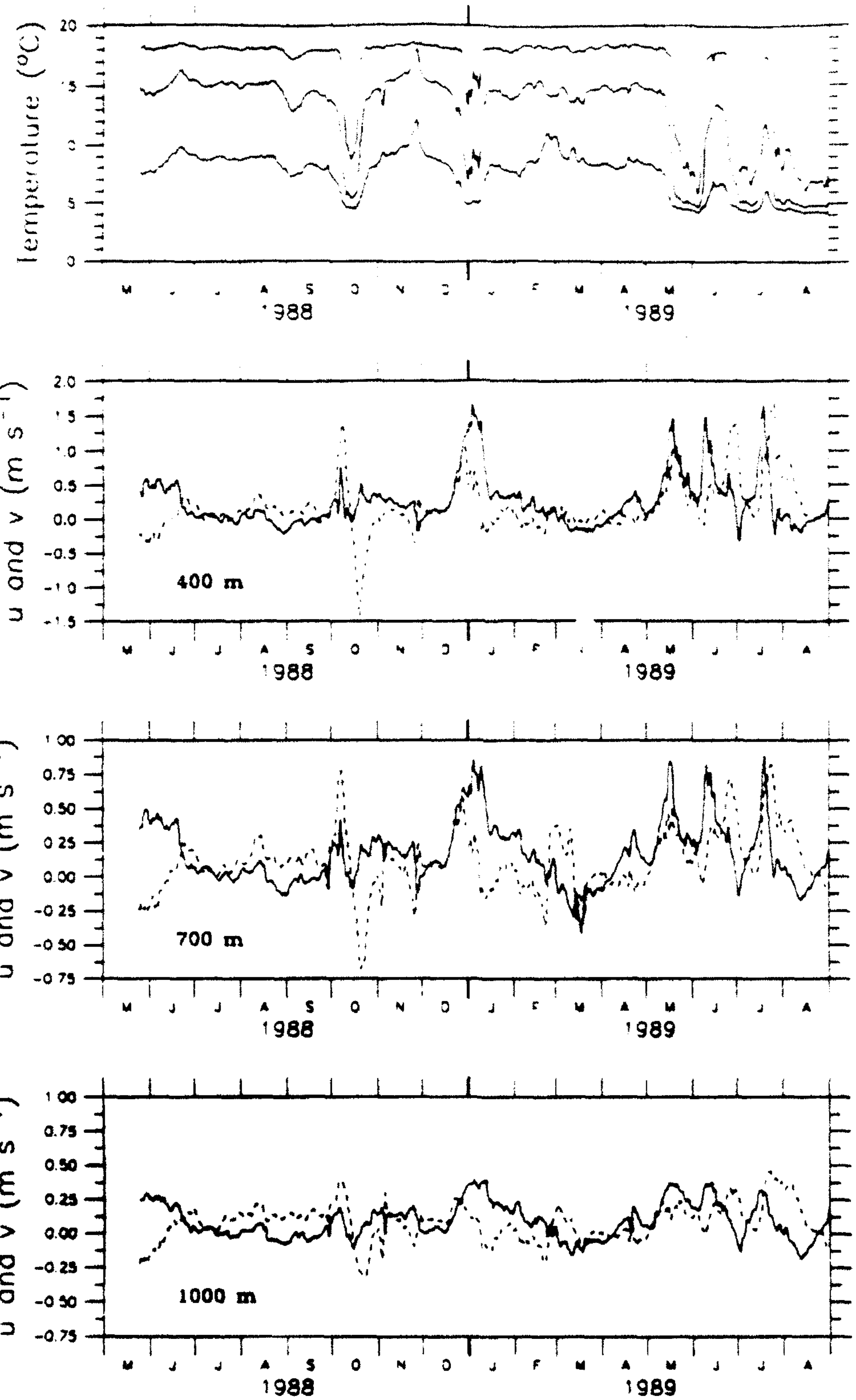
Site I4 Year 1



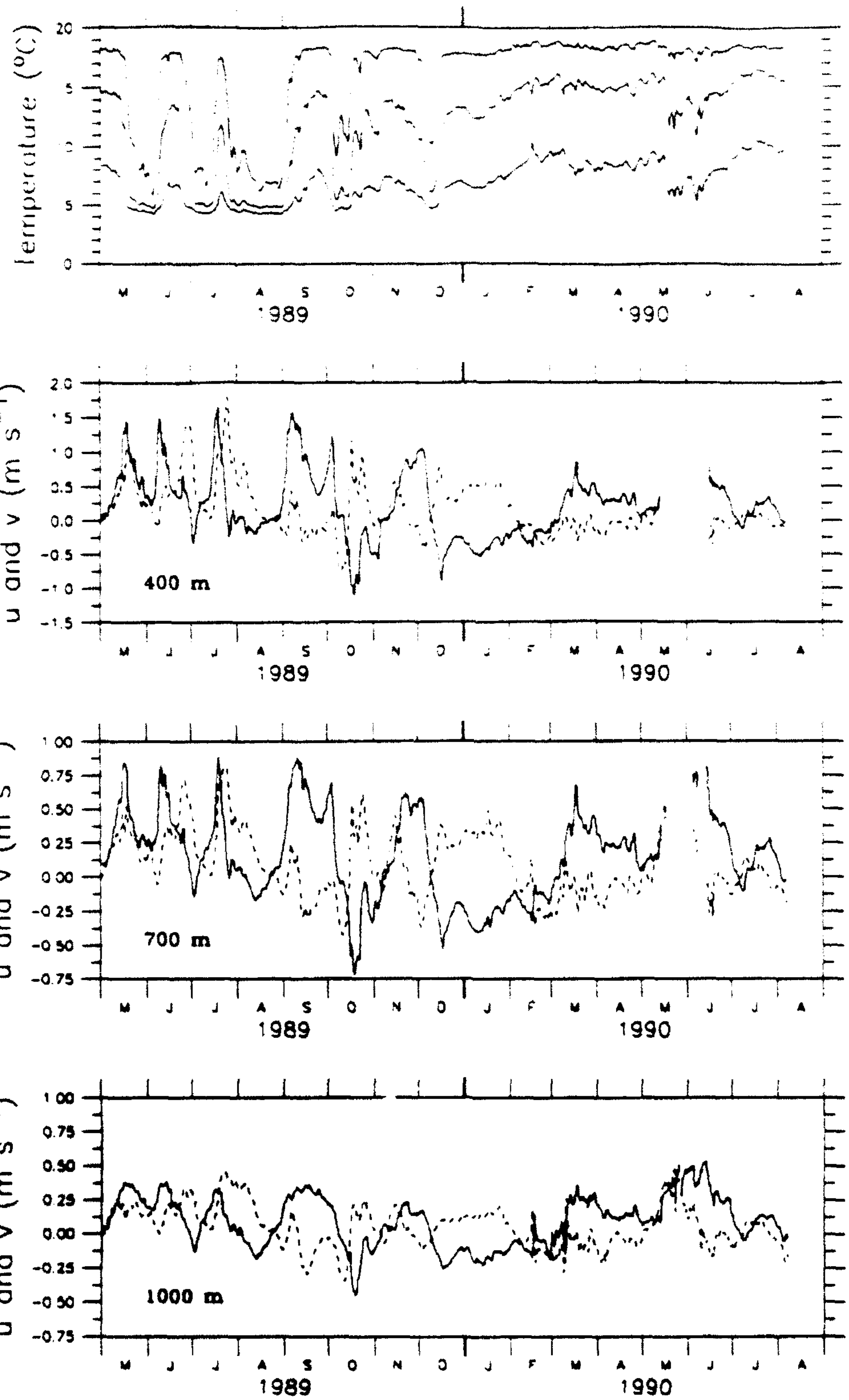
Site I4 Year 2



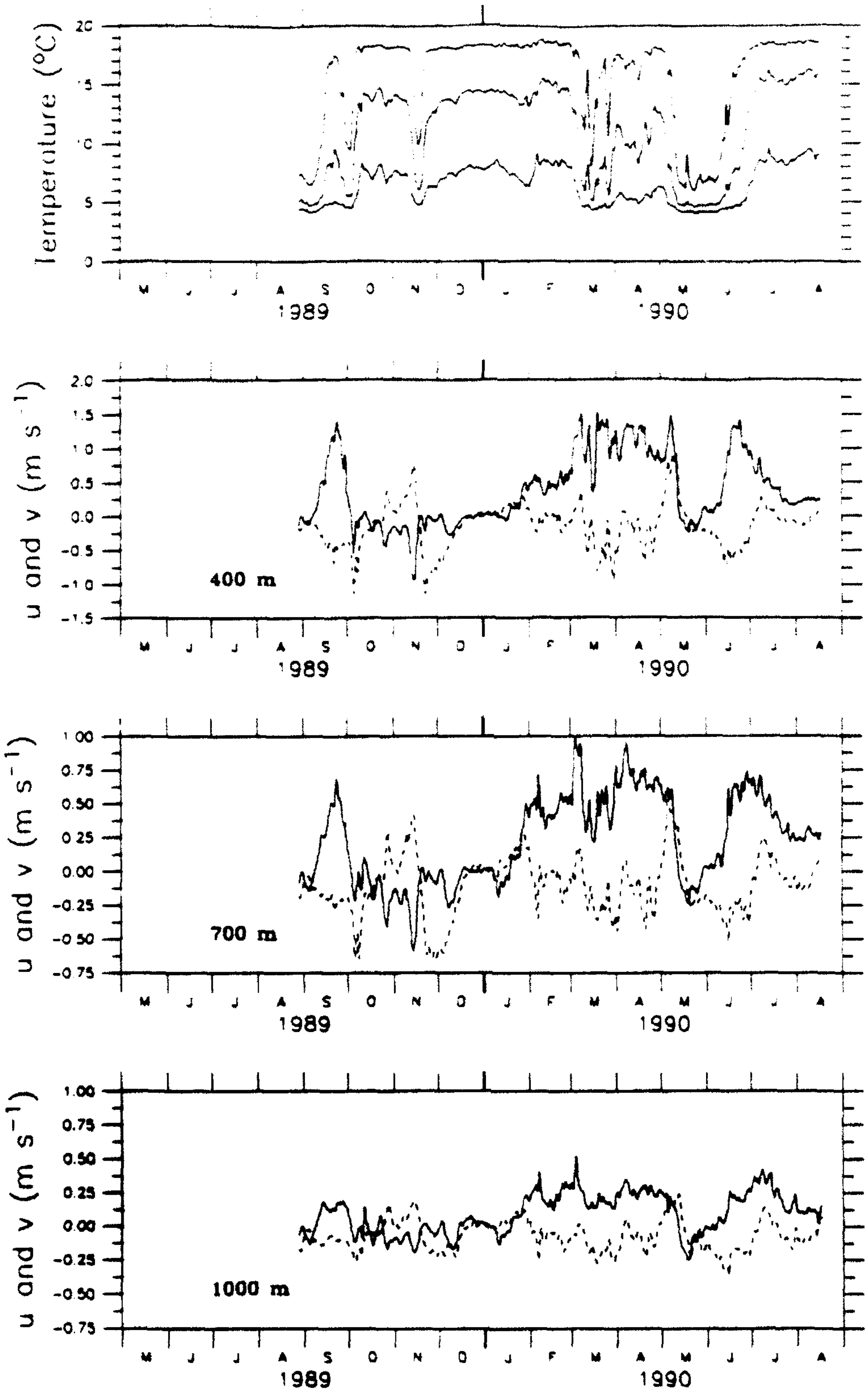
Site I5 Year 1



Site I5 Year 2



Site M13 Year 2



SECURITY CLASSIFICATION OF THIS PAGE

REPORT DOCUMENTATION PAGE

1a. REPORT SECURITY CLASSIFICATION Unclassified	1b. RESTRICTIVE MARKINGS
2a. SECURITY CLASSIFICATION AUTHORITY	3. DISTRIBUTION/AVAILABILITY OF REPORT Distribution for public release; Distribution is unlimited.
2b. DECLASSIFICATION/DOWNGRADING SCHEDULE	
4. PERFORMING ORGANIZATION REPORT NUMBER(S) University of Rhode Island Graduate School of Oceanography GSO Technical Report 92-4	5. MONITORING ORGANIZATION REPORT NUMBER(S)
6a. NAME OF PERFORMING ORGANIZATION University of Rhode Island Graduate School of Oceanography 6b. OFFICE SYMBOL (If applicable) 1122 PO	7a. NAME OF MONITORING ORGANIZATION
6c. ADDRESS (City, State, and ZIP Code) South Ferry Road Narragansett, RI 02882-1197	7b. ADDRESS (City, State, and ZIP Code)
8a. NAME OF FUNDING/SPONSORING ORGANIZATION Office of Naval Research National Science Foundation 8b. OFFICE SYMBOL (If applicable)	9. PROCUREMENT INSTRUMENT IDENTIFICATION NUMBER N00014-90J-1568, N00014-90J-1548 OCE97-17144
8c. ADDRESS (City, State, and ZIP Code) 800 N. Quincy Street, Arlington, VA 22217 1800 G. Steet, NW, Washington, DC 20550	10. SOURCE OF FUNDING NUMBERS PROGRAM ELEMENT NO. | PROJECT NO. | TASK NO. | WORK UNIT ACCESSION NO.

11. TITLE (Include Security Classification)
Mooring Motion Correction of SYNOP Central Array Current Meter Data

12. PERSONAL AUTHOR(S) Meghan Cronin, Karen Tracey and D. Randolph Watts

13a. TYPE OF REPORT	13b. TIME COVERED	14. DATE OF REPORT (Year, Month, Day)	15. PAGE COUNT
summary	FROM 5/88 TO 8/90	March 1993	115

16. SUPPLEMENTARY NOTATION

17. COSATI CODES			18. SUBJECT TERMS (Continue on reverse if necessary and identify by block number)
FIELD	GROUP	SUB-GROUP	Gulf Stream, SYNOP, Current Meter Mooring Corrections

19. ABSTRACT (Continue on reverse if necessary and identify by block number)

From May of 1988 to August 1990, as part of the SYNOP field program, twelve tall moorings measured the Gulf Stream's temperature and velocity fields at nominal depths of 400 m, 700 m, 1000 m, and 3500 m. Although stiff, high-performance moorings were used to maintain the top current meters at approximately 400 m below the surface (~ 4000 m above the sea floor), the jet's drag caused the moorings to make vertical excursions.

Therefore, the current meter data were corrected to constant horizons using a modified version of Hogg's (1991) mooring motion correction scheme. An important extension of Hogg's (1991) method is the inclusion of a weighted interpolation of the measured temperatures. This modification assures that as the current meter measurements approach the respective nominal depths, the corrected temperature and velocity outputs smoothly approach the measurements; i.e. the compensated u,v,T records are truer to the measured records.

This report documents the mooring motion correction of the SYNOP Central Array temperature and velocity data.

20. DISTRIBUTION/AVAILABILITY OF ABSTRACT ☒ UNCLASSIFIED/UNLIMITED ☐ SAME AS RPT ☐ DTIC USERS	21. ABSTRACT SECURITY CLASSIFICATION
22a. NAME OF RESPONSIBLE INDIVIDUAL D. Randolph Watts	22b. TELEPHONE (Include Area Code) | 22c. OFFICE SYMBOL

DD FORM 1473, 84 MAR   83 APR edition may be used until exhausted. All other editions are obsolete.   SECURITY CLASSIFICATION OF THIS PAGE